THE JOSHUA SYSTEM

VOL. 4 LATTICE PHYSICS

Approved by

H. C. Honeck, Research Manager
Computer Applications Division

NOTICE

This document contains descriptions of the JOSHUA Operating System and Applications Modules. The document is prepared as a set of loose-leaf notebooks which will be periodically updated. ~~_____~~
~~_____~~

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

June 1970

E. I. DU PONT DE NEMOURS & COMPANY
SAVANNAH RIVER LABORATORY
AIKEN, S. C. 29801

MASTER

CONTRACT AT(07-2)-1 WITH THE
UNITED STATES ATOMIC ENERGY COMMISSION

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

EB

LEGAL NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

Lattice Physics

Part A	Subsystem Overview
Part B	Data Management
Part C	Geometry Modules
Part D	Nuclear Data Modules
Part E	Integral Transport Modules

JOSHUA DPSTM-500	Title Part A - Subsystem Overview	Vol. IV	Page A.O.1
		Rev.	Date 6/70

Contents

- A.1 Overview of the Lattice Physics Subsystem
- A.2 Lattice Geometry and Materials

JOSHUA DPSTM-500	Title Part A - Subsystem Overview	Vol. IV	Page A.O.2
		Rev.	Date 6/70

(Blank)

Work done by

H. C. Honeck

Report written by

H. C. Honeck

Contents

	<u>Page</u>
1. Introduction	A.1.3
2. Role of RAHAB in Reactor Design	A.1.3
3. Review of the Basic Theory	A.1.6
4. Module and Data Set Organization	A.1.12

JOSHUA DPSTM-500	Title Overview of the Lattice Physics Subsystem	Vol. IV	Page A.1.2
		Rev.	Date 3/70

(Blank)

1. Introduction

The series of modules and paths in JOSHUA concerned with the calculation of the neutron distribution in a reactor lattice is called the lattice physics subsystem. From the user's point of view, the subsystem is a single module called RAHAB. This module is a control module (or path) which executes many other modules. From the programmer's point of view the subsystem is a large collection of modules which interface through a temporary data set called the Lattice Intermediate Data Set (LIDS).

This document, and the remainder of the documents in this series, are written for the benefit of programmers who will be modifying the modules in the future; user's wanting a detailed knowledge of the theory, approximations, and operation; and for off-site programmers who may have to adapt RAHAB to their own computer environment. A second document will be prepared which will be more of a "Users Manual".

In the remainder of this document, we first show how RAHAB fits into the over-all reactor design capabilities of JOSHUA. Next, we briefly outline the basic theory used in RAHAB, and finally describe the organization of the program into functional modules.

2. Role of RAHAB in Reactor Design

The design of heterogeneous reactors is traditionally divided into several distinct stages characterized by the detail used to describe the neutron energy spectrum and spatial distribution. In the first stage, little or no spatial detail is included and neutron spectra are computed with high energy resolution (hundreds to thousands of groups). These spectra are used as weighting functions to average cross sections and compress the group structure (tens to hundreds of groups).

This process is called "Basic Cross Section Library Preparation" in Figure 1. In this figure circles represent JOSHUA data sets permanently stored on disk storage. The rectangles represent JOSHUA modules or collections of modules. The national point cross section file (ENDF/B) is converted to group-averaged cross sections and stored on the FINEGRP and NUCPARAM data sets. The module JMGP1 can also be used to translate and store existing HAMMER 84 group libraries on FINEGRP. These data can then be converted to the MULTIGRP data set (CØFIN) and compressed (PRESS) using spectra produced by B_L calculations.

Figure 1. Relationship of RAHAB with Other JOSHUA Modules and Data Sets.

The second stage of reactor design is called "Lattice Fine-Structure Calculation" in Figure 1. Attention here is focused on a small portion of the reactor (a cell or supercell) comprising some 10 to 100 regions. Descriptions of these cells are contained on the ASSEMBLY library. The neutron population can vary markedly from one region to another. This variation is called the "fine-structure" and full multigroup transport theory is required to predict it. The HAMMER code was developed to calculate fine-structure in simple cells, and RAHAB developed for complex supercells. These calculations currently use 84 energy groups.

Clearly, one cannot calculate the fine-structure simultaneously in hundreds of different cells so that some simplification is required. Since fine-structure is primarily a local effect and not strongly influenced by broad flux tilts in the reactor, it is generally assumed that adequate results are obtained by analyzing a single cell or supercell in an infinite environment of identical cells or supercells. This then allows the use of simple conservative boundary conditions on the one cell or supercell to be analyzed.

Resonance capture calculations actually fall somewhere between the first and second stages of this picture in that thousands of energy groups and a very few (2 - 5) spatial regions are used. These transport theory calculations are a part of RAHAB and use resonance parameters from the NUCPARAM data set and cell descriptions from the ASSEMBLY data set. The resulting resonance reaction rates are stored on the RILIB data set.

Once the fine-structure has been calculated, it can be used to homogenize the cell or supercell, that is, convert the heterogeneous cell to a uniform paste with properties that preserve all neutron reaction rates. This is usually done in terms of a 4-group cross section set (FEWGRP data set).

The third and later stages of reactor design called "Full Reactor Static and Kinetic Calculations" in Figure 1 use the few-group cross sections sets in diffusion theory calculations to obtain fluxes and reaction rates in the entire reactor consisting of hundreds of cells.

The various stages in reactor design sketched above are dictated primarily by physical considerations and experience. At each stage the amount of detail that can be treated depends on calculational considerations. It is interesting to note that the product of the number of energy groups and the number of spatial region (or flux points) is roughly the same in each stage. This constant is about 1000 - 2000 for computers of the IBM 704 and IBM 7090 class, and 5000 - 10,000 for the IBM 360, CDC 6600, and UNIVAC 1108 class.

3. Review of the Basic Theory

The basic theory used in RAHAB is the same as used in THERMOS and HAMMER, that is, multigroup integral transport theory. No extensions or improvements of this theory have been made in RAHAB. However, many improvements have been made in the numerical procedures used to solve the integral transport equation.

A multigroup Monte Carlo capability is also being developed. While Monte Carlo calculations are too expensive for routine production use, they are essential to evaluate the main theoretical approximation used in RAHAB (isotropic scattering in the transport approximation) and the numerical approximations used to evaluate transport kernels and decouple the supercell into current-coupled zones. A review of the Monte Carlo method will not be given here.

A detailed treatment of the methods used to solve the integral transport equation will be given in later sections. In this section only brief statements of the equations are given to provide the background required to understand the subdivision of RAHAB into separate functional modules interfaced to one another through temporary data sets.

We consider a two-dimensional space assumed to be uniform in the third dimension. The symbol $\underline{r}$ denotes a point in that space and $d\underline{r}$ denotes a differential volume (area times 1 cm in the third dimension) element. Neutron energy is denoted by E . Consider a mono-energetic isotropic line source located at $\underline{r}'$ which emits one neutron/sec/cm of length. Neutrons from this source travel in all directions through the space until they make a collision with an atom. Temporarily we consider them to be lost once they have made a collision. Later, we will reintroduce them as a collision source term. The flux of neutrons of energy E at another point $\underline{r}$ is the number of uncollided neutrons passing through a unit area at $\underline{r}$.

The uncollided flux at $\underline{r}$ owing to the unit source at $\underline{r}'$ is easily calculable from the basic laws of neutron transport and is denoted by $T(\underline{r}, \underline{r}', E)$. If we have a distribution of isotropic sources, $H(\underline{r}', E)$, then the flux $\phi(\underline{r}, E)$ is the sum (integral) of the individual contributions.

$$\phi(\underline{r}, E) = \int d\underline{r}' T(\underline{r}, \underline{r}', E) H(\underline{r}', E) \quad (1)$$

Since $T(\underline{r}, \underline{r}', E)$ appears as the kernel of the integral transport equation, it is called the "transport kernel".

Next we consider the various terms that make up the source $H(\underline{r}', E)$. First there may be some fixed external sources denoted by $H_0(\underline{r}', E)$. Second, there are the neutrons produced by fission. The number of neutrons produced from fission in isotope k at the point $\underline{r}'$ is

$$F_k(\underline{r}') = \frac{1}{k_{\text{eff}}} \int_0^{\infty} dE' \nu_k(\underline{r}', E') \Sigma_{fk}(\underline{r}', E') \phi(\underline{r}', E') \quad (2)$$

where k_{eff} is the effective multiplication (or eigenvalue). These neutrons are emitted in a normalized fission spectrum, $\chi_k(E)$, so that the total fission source is

$$H_1(\underline{r}', E) = \sum_k \chi_k(E) F_k(\underline{r}') \quad (3)$$

The third (and largest) source results from collisions of neutrons with atoms.

$$H_2(\underline{r}', E) = \int dE' \Sigma_s(\underline{r}', E, E') \phi(\underline{r}', E') \quad (4)$$

The term $\Sigma_s(\underline{r}', E, E')$ is called the "scattering kernel" and describes the change in energy of a neutron when it makes a collision with an atom. Neutrons from the (n,2n) reaction are also included in this term. The total source in equation (1) is

$$H(\underline{r}', E) = H_0(\underline{r}', E) + H_1(\underline{r}', E) + H_2(\underline{r}', E) \quad (5)$$

and we note that equation (1) is an integral equation since the flux appears both on the left-hand side and in the integrals on the right-hand side.

The method for solving these equations proceeds as follows. First, the energy range is subdivided into intervals called groups. The approximation is made that

$$\int dE f(E) = \sum_j \Delta E_j f_j \quad (6)$$

where ΔE_j is the width of the j^{th} group, and f_j is the average value of a function $f(E)$ in the group. Similarly, the space

is subdivided into volume elements called sub-regions. It is assumed that

$$\int d\mathbf{r} f(\mathbf{r}) = \sum_n V_n f_n \quad (7)$$

where V_n is the volume of the n^{th} sub-region, and f_n is the average value of a function $f(\mathbf{r})$ in the sub-region.

The second step is to convert equations (1-4) into coupled sets of linear equations using approximations (6-7).

$$\phi(n,j) = \sum_m V(m) T(n,m,j) H(m,j) \quad (8)$$

$$F_k(m) = \frac{1}{k_{\text{eff}}} \sum_i \Delta E(i) v_k(m,i) \Sigma_{fk}(m,i) \phi(m,i) \quad (9)$$

$$H_1(m,j) = \sum_k x_k(j) F_k(m) \quad (10)$$

$$H_2(m,j) = \sum_i \Delta E(i) \Sigma_s(m,j,i) \phi(m,i) \quad (11)$$

If the lattice is divided into many sub-regions, the matrix $T(n,m,j)$ will be very large. It is convenient to divide the space into "zones" and call the boundaries of the zones "faces". If the index m numbers the collection of sub-regions and faces in a zone, equation (8) is written as

$$\begin{aligned} \phi(n,j) = & \sum_m V(m) T^{vv}(n,m,j) H(m,j) \\ & + \sum_m 4S(m) T^{vs}(n,m,j) J^{\text{in}}(m,j) \end{aligned} \quad (12)$$

$$J^{\text{out}}(n,j) = \sum_m V(m) T^{\text{sv}}(n,m,j) H(m,j) + \sum_m S(m) T^{\text{ss}}(n,m,j) J^{\text{in}}(m,j) \quad (13)$$

where $J^{\text{in}}(m,j)$ and $J^{\text{out}}(m,g)$ are currents through face m of area $S(m)$ going into and out of the zone. The boundary conditions relate these currents.

$$J^{\text{in}}(N,j) = \sum_m B(N,M) J^{\text{out}}(M,j) \quad (14)$$

where N, M number faces the supercell. The use of current-coupled zones makes large problems tractable at the expense of the accuracy of the transport matrix.

The treatment of resonance capture follows that of HAMBUR which is an extension of the HAMMER treatment. Briefly, a simplified form of the integral transport equation is solved for each resonance assuming that the resonances are widely separated so that an asymptotic flux exists above the resonance energy. Further it is assumed that interaction between cells is small so that resonance reaction rates are computed for each cell in a uniform lattice. Spatial distributions of resonance rates within a cell are calculated and these rates depend on the total slowing-down density in each cell.

The resonance absorption and fission rates in isotope k , group j , sub-region m , based on a unit total slowing-down source in the cell are denoted by $R_{ak}(m,j)$ and $R_{fk}(m,j)$. The total number of neutrons in cell c slowing-down past energy $E(j)$ is given by

$$Q(c,j) = \sum_{m \in c} V_m \sum_{i < j} \Delta E(i) \phi(m,i) \sum_{l \geq j} \Delta E(j) \Sigma_s(m,l,i) \quad (15)$$

The total reactions in group j , isotope k , and sub-region m are then $R_{ak}(m,j) Q(c,j)$ and $R_{fk}(m,j) Q(c,j)$. The fission contributions are added to equation (9). The fraction of the slowing-down source in the cell which escapes resonance capture is

$$p(c,j) = 1 - \sum_{m \in c} \sum_k R_{ak}(m,j) \quad (16)$$

The source term, equation (11), is then reduced to account for resonance capture.

$$H_2^*(m,j) = H_2(m,j) \left\{ 1 - \frac{[1 - p(c,j)] Q(c,j)}{\sum_{m \in c} V(m) H_2(m,j)} \right\} \quad (17)$$

The resulting set of matrix equations (8-17) are solved by an iterative procedure as follows:

1. Establish a spatially uniform flux proportional to I/E in the energy region containing the fission spectrum.
2. Solve for the flux in each epithermal group starting below the fission spectrum. Since each group depends only on higher energy groups, only spatial iteration is required.
3. Compute the slowing-down source to the thermal region and guess a thermal flux distribution.
4. Iterate the spatial flux distribution for each thermal energy group.
5. Repeat step 4 until the energy distributions have converged.
6. Compute the fission source from the epithermal and thermal flux.
7. Iterate the spatial distribution for each fast energy group and calculate the first fission source.
8. Repeat step 7 until the fast flux has converged.
9. Repeat steps 2-5.

The boundary conditions used in the above calculations conserve neutrons so that there is no leakage of neutrons from the supercell. The B_L method is used to predict the change in energy spectrum caused by leakage characterized by a buckling, B^2 . It is assumed that leakage has a negligible effect on the spatial distributions within the supercell and that the major effect is a smooth tilting of the energy spectrum. Thus, the first step is to homogenize the supercell using the calculated flux and slowing-down density as weighting functions. For example, the homogenized absorption cross section is given by

$$\Sigma_a(j) = \sum_n V(n) \Sigma_a(n) \phi(n,j) / \sum_n V(n) \phi(n,j) \quad (18)$$

The homogenization formulas insure that all reaction rates are preserved when the B_ℓ equation is solved at a $B^c = 0$.

The B_ℓ equations are obtained from the Fourier Transform of the one-dimensional multigroup transport equation. They can be written in the form

$$\Sigma(j) \phi_\ell(j) = H_\ell(j) + \sum_{n=0}^1 R_{\ell n}(B,j) H_n(j) \quad ; \quad \ell=0,1,2,3 \quad (19)$$

$$H_n(j) = \sum_i \Delta E(i) \tau_s(j,i) \phi_n(j) + \delta_{no} \sum_k \chi_k(j) F_k \quad (20)$$

$$F_k = \sum_i \Delta E(i) v_k(i) \tau_{fk}(i) \phi_0(i) \quad (21)$$

$$R_{\ell n}(B,j) = (iB)^{n-\ell} \left[\frac{2n+1}{2} \Sigma(j) \int_{-1}^1 d\mu \frac{P_\ell(\mu) P_n(\mu)}{\Sigma(j) - i\mu B} - \delta_{\ell n} \right] \quad (22)$$

In these equations the cross sections and scattering matrices are those obtained by homogenization. The subscripts, ℓ, n are Legendre indices resulting from the expansion of the angular flux in Legendre polynomials, $P_n(\mu)$. These equations are modified for resonance capture in the same manner as equations (15-17).

These equations are solved in the following manner:

1. Use only $\chi(j)$ for ^{235}U for the second term in (19).
2. Solve the epithermal equations.

3. Solve the thermal equations by iterating on the energy distribution.
4. Compute the fission terms, F_k , and the second term in (19).
5. Repeat from step 2 until the F_k converge.
6. If the critical buckling was requested, repeat from step 2 varying B^2 until $k_{eff} = 1$.

The resulting scalar flux, $\phi_o(j, B^2)$, is used to calculate a "leakage corrected" fine-structure flux.

$$\phi^*(n, j, B^2) = \phi_o(j, B^2) \phi(n, j) / \sum_n V(n) \phi(n, j)$$

which is used to compute reaction rates and few-group cross section sets.

4. Module and Data Set Organization

A flow diagram for RAHAB is shown in Figure 2. RAHAB consists of:

- Calculation modules
- JOSHUA permanent data sets
- LIDS temporary storage

As shown in Figure 1, RAHAB is part of the over-all reactor design capabilities of JOSHUA. Communication between RAHAB and other design modules is via the permanent JOSHUA data sets named ASSEMBLY, NUCPARAM, MULTIGRP, RILIB, LATTICE, and FEWGRP. These data sets are described in Volume III. The first three data sets are the input to RAHAB and the last three contain the RAHAB output of lasting value.

There are many reasons for writing RAHAB as a series of functional modules rather than one large code. The main reasons are:

- Ease of code development and maintenance.
- Smaller core storage requirements.
- Flexibility to prepare several control programs
- Usefulness of modules for other purposes.

Figure 2. RAHAB Modules (rectangles) and Data Sets (circles and ovals). Solid lines represent data flow. Dashed lines show sequence of module execution. RAHAB control program not shown.

Two illustrations of the last point are that 7 of the 9 RAHAB modules can be used directly for a Monte Carlo version, and 4 can be used to provide a stand-alone B_c calculation for a homogeneous medium.

The modules communicate through the Lattice Intermediate Data Set (LIDS). The JOSHUA data manager is used to read and write this storage and, from the viewpoint of the modules, it is simply another JOSHUA data set. The differences between the permanent data sets and the LIDS are:

- The LIDS cannot be permanently saved (but is retained throughout each step of a multi-job session).
- The LIDS cannot be displayed at a terminal because of the large record size and list-processed structure of the records.

The LIDS consists of a group of well-defined record types. Descriptions of these record types are given in Manuscript IV.B.2.

The RAHAB modules are written in Fortran IV and all use a similar COMMON layout. While they are used as modules at the present time in the JOSHUA system, they can be used in an overlay structure. Each module can be executed in a 240K byte (60K word) partition. A functional description of each module is given below.

RAHAB - RAHAB is a control module whose main functions are to establish data set names, search options, and edit options, and pass these on to all modules. RAHAB executes each module in sequence as shown in Figure 2.

The present control module allows for multiple cases but does not attempt to utilize results from one case to speed up a later case. Other versions will make use of previous results and will control material or composition searches.

ASP - Assembly (cell) descriptions are given on the ASSEMBLY data set in terms of physical regions as they would appear on an engineering drawing. Compositions are given in units convenient to the user. RASP takes the descriptions of individual cells, modifies the geometric description based on input sub-division or homogenization specifications, changes dimensions to cm, changes concentration units to atoms/barn-cm, and combines the cells into a supercell. Boundary conditions are built into tables in the LATTICE data set so that the user need refer only to a predefined

lattice pattern. The results are written in a record which imitates a deck of cards.

LAGER - LAGER takes the geometric and material description of the supercell from the ASP module and converts them to detailed packed tables needed by Monte Carlo tracking routines. These tables describe the geometry by vertices (points) connected by straight lines or circular arcs. Pointer variables are used to express the relationship between elements of the geometry and between the geometry and materials. Zone tables are also constructed which describe how the transport kernels are to be segmented into full matrices or surface-coupled matrices.

Unusual geometries which cannot be handled by ASP can be input by constructing an input record from cards. This feature is primarily of use in Monte Carlo calculations.

CREEP - Geometric descriptions from the ASP module and resonance parameters from the NUCPARAM data set are used to compute resonance capture and fission rates for each isotope, resonance, and region of a cell. It is assumed that each cell is in an infinite uniform lattice of cells of the same type, and that there is no overlapping of resonances. The results for each cell are permanently stored on the RILIB data set.

RESIN - Resonance rates by isotope, region, and resonance energy are read from the RILIB data set and converted to rates by isotope, region, and energy group. Since there will be many resonances in a group, precalculation of the group rates will reduce later computing time. Corrections to the resonance rates from small changes in concentrations or temperature are made so that the complete resonance calculation may not have to be repeated for burnup or kinetic studies.

CRISP - Cross sections, scattering matrices, and fission spectra are read from the MULTIGRP data set and packed into blocks. Based on the problem size, the energy range is divided into bands so that the iteration module FIST can contain the data for one band in core. Data blocks are then written on the LIDS by energy band.

These functions were placed in a separate module so that they could be by-passed for parts of either a multi-case job or a concentration search. Note that most of the RAHAB input/output time is used to read data from MULTIGRP.

- TRAMP - TRAMP computes and writes the transport matrices for each zone and energy band. The calculation may be done using the cosine current, direct integration, or statistical integration (Monte Carlo) method. (The cosine current method is implemented only for annular geometry.)
- Input to TRAMP indicates what zones are to be done so that, in a concentration search, unnecessary recalculation can be avoided.
- FIST - FIST solves the integral transport matrix equations by iteration. The calculation starts just below the fission spectrum, proceeds down through the epithermal and thermal ranges, and computes the fission distribution. The fast region is then iterated to settle the fast fission sources and the epithermal and thermal regions recalculated.
- PASTE - Cross sections, scattering matrices, and resonance rates are homogenized using the fine-structure flux and slowing-down density from FIST. If cross section data is not found in the LIDS, the data is read from MULTIGRP. If flux or slowing-down density data is not found, a spatially flat distribution is assumed. These options are provided so that PASTE can be used as part of a homogeneous medium B_L calculation.
- BLEAK - The multigroup B_L equations are solved by iteration. Search options are:
- k_{eff} calculation
 - buckling search
 - time eigenvalue (α) search

The resulting Legendre fluxes (P_0 to P_3) and results for each search are written on the LIDS.

JOSHUA DPSTM-500	Title Overview of the Lattice Physics Subsystem	Vol. IV	Page A.1.17
		Rev.	Date 3/70

RABED - Fluxes and currents are corrected for leakage based on the results of the B_0 calculation. Few group cross sections by cell and isotope are computed and written on the FEWGRP data set. Fluxes, currents, and selected integral results are written on the LATTICE data set. A variety of edits are produced and printed.

Work Done By

W. H. Hightower
H. C. Honeck
C. E. Parker

Report Written By

H. C. Honeck

Contents

	<u>Page</u>
1. Introduction	A.2.2
2. Elements of the Geometry	A.2.2
2.1 Coordinate System	A.2.2
2.2 Vertices, Radial Arcs, and Sides	A.2.3
2.3 Sub-regions and Regions	A.2.3
2.4 Sub-assemblies, Faces, and Zones	A.2.6
2.5 Cells, Supercells, and Lattice Positions	A.2.8
3. Boundary Conditions	A.2.8
3.1 Basic Formulae	A.2.8
3.2 Hexagonal Lattices	A.2.10
3.3 Square Lattices	A.2.13
4. Input Preparation Facilities	A.2.15
4.1 ASSEMBLY Data Set	A.2.15
4.2 Execution-time Modification	A.2.15
4.3 Lattice Patterns	A.2.16

JOSHUA

DPSTM-500

Title Lattice Geometry and Materials

Vol.

IV

Page

A.2.2

Rev.

Date

4/70

(Blank)

1. Introduction

Both Monte Carlo and integral transport methods are used to compute neutron flux distributions in reactor lattices. Since the Monte Carlo method can treat complex, 3-dimensional geometry, it requires a very complex and detailed description of the geometry. Integral transport calculations have been used for 1-dimensional or simple 2-dimensional configurations in the past and only a simple geometrical description was required. Integral transport calculations for realistic 2-dimensional configurations require the same complex geometrical description required by the Monte Carlo calculations so that it is well to start with a description which is usable by both methods. This has two advantages; the Monte Carlo tracking routines can be used to generate transport kernels, and a full Monte Carlo calculation can be used to evaluate the approximations involved in the integral transport method. For these reasons, the geometrical descriptions in the next two sections are those required for Monte Carlo calculations.

It would be quite difficult for the user to input geometrical descriptions in the detail required internally by the programs. Geometrical descriptions in user-oriented terms are catalogued on the ASSEMBLY data set and facilities have been provided to expand these into the detailed descriptions. These facilities will be described in Section 4.

2. Elements of the Geometry

Most lattice configurations are well represented by a 2-dimensional space. Further, nearly all practical configurations can be represented as area elements bounded by straight lines or circular arcs. Therefore we assume that a lattice is uniform and infinite in the z direction and consider only projections or cuts on the x-y plane. We will still refer to the "volume" of an element and mean the area in the x-y plane times one cm. in the z direction. The "area" of a surface will mean the length of a line in the x-y plane times one cm. in the z direction.

We could organize the following material in two ways; start with complex elements and work down to elementary elements, or vice versa. The first way is easier for the reader, the second is easier for the writer. Thus, we have chosen the second way and start with the elementary elements.

2.1 Coordinate System

The coordinate system used is illustrated in Figure 1, but, as previously mentioned, we will be mainly concerned with the x-y plane. The coordinate system is assumed to be centered in a cell of the lattice. As a particle leaves one cell and enters another, its coordinates are

changed to refer to the coordinate system of the cell entered. These coordinate changes are discussed under Boundary Conditions.

2.2 Vertices, Radial Arcs, and Sides

A vertex is a point in the x-y plane. A radial arc is a segment of a circle lying between two vertices. Since two vertices on a circle separate the circle into two arcs, we adopt the convention that the radial arc of interest is the one obtained by traveling counter-clockwise (increasing θ) from the first vertex listed to the second vertex listed. If no vertices (or one vertex) are given, the radial arc is the entire circle.

A side is a line or radial arc connecting two vertices. Again, in the special case where the side is a full circle, the vertices may be omitted.

These definitions are illustrated in Figure 2.

2.3 Sub-regions and Regions

The shaded area in Figure 2 is called a sub-region. It is defined as a portion of the x-y plane enclosed by sides. The sub-region illustrated is enclosed by three straight-line sides, one radial arc, and one full circle. Any combination (number and type) of sides may be used provided that the entire space of interest is filled with sub-regions. Thus a side separates two sub-regions. If the side is a radial arc, the sub-region lying inside the circle is listed first, and the sub-region lying outside the circle listed second. Special sides (called faces) will be defined later to treat boundaries.

A collection of contiguous sub-regions all containing the same material is called a region. The shaded area in Figure 3 illustrates an annular region consisting of sixteen sub-regions.

A region need not be subdivided into sub-regions (a region may consist of only one sub-region). Since a region is uniform in its properties, it is used to specify materials and edit reaction rates and flux. Sub-regions are used to obtain more detailed edits (Monte Carlo calculations) and to properly account for flux distributions within a region (integral transport calculations).

Fig. 1

Fig. 2

Fig. 3

2.4 Sub-assemblies, Faces, and Zones

A sub-assembly is a collection of contiguous regions of the same geometric type. Nested annuli, nested hexagons, and clusters are examples of sub-assemblies (examples of sub-assemblies are shown in Figure 4).

The term "assembly" would logically be defined as a collection of contiguous sub-assemblies. However, to avoid confusion with the engineer's definition of assembly, and because it is not required, the term will not be used.

A sub-assembly can be specified independently of the sub-assemblies which surround it. The external sides of one sub-assembly need not be the same as those of the adjacent sub-assembly, although each set of sides must cover the mutual surface. These external sides of a sub-assembly are called faces. The boundary between two sub-assemblies is illustrated in Figure 5. For clarity, a gap is shown between sub-assemblies. Actually no gap is allowed and there is just one boundary surface. Each side which lies on the boundary is called a face, and the boundary conditions (which will be described later) specify the linkages between faces.

It is desirable for integral transport calculations to define collections of sub-assemblies in which a common transport theory is used, or where the transport matrix can be approximated by weakly coupled sub-matrices. For this purpose a zone is defined as a collection of contiguous sub-assemblies. Zones are not required for Monte Carlo calculations.

Fig. 4

Fig. 5

2.5 Cells, Supercells, and Lattice Positions

A cell is a collection of contiguous sub-assemblies and occupies the area within a hexagonal or rectangular boundary. All cells of a lattice must be of the same over-all size and shape so that the cells can fill all space. A hexagonal cell is illustrated in Figure 4.

A supercell is the smallest collection of contiguous cells which repeats itself by translation to fill all space and for a lattice. If there is only one cell type (uniform lattice), the supercell consists of only one cell. If there are several cell types (mixed lattice), the supercell consists of several cells, one for each different type of cell in each of its possible symmetry positions. An example of supercell is shown in Figure 6 as the 7 cells inside the heavy boundary line.

A lattice position is an arbitrary number assigned to a cell in a supercell. For example, in Figure 6, we might assign the central cell lattice position 1, the cell to the right of the central cell lattice position 2, and the remaining cells (going counter-clockwise) lattice positions 3 - 7. We would then say that cell type C was in lattice position 1 and cell type D in lattice position 2 - 7. The flux in cell type D is the same in all lattice positions because of the symmetry of the lattice so that only the shaded area need be explicitly represented.

3. Boundary Conditions

When a particle leaves any sub-assembly through a face, it enters another sub-assembly. If both sub-assemblies belong to the same cell (and hence have the same coordinate origin), the position and direction of the particle are not altered. When the sub-assembly is the last sub-assembly of a cell, the particle crosses a cell boundary and enters an adjacent cell whose coordinate system is translated and rotated with respect to the coordinate system of the cell from which the particle left. The boundary conditions specify the change in the particle's coordinates and direction which occur on crossing a face.

3.1 Basic Formulae

Consider the particles labeled 1, 2, 3 in Figure 6 which cross cell boundaries. Particle 1 leaves cell type D and enters the adjacent D. Translation of the supercell shows that this is equivalent to a particle entering the supercell at 1'. Rotation of the supercell shows that this equivalent to entering the original cell at 1". In a similar manner, translation moves the particle from 2 to 2', and rotation from 2' to 2". Only rotation is required for particle 3.

Fig. 6

Fig. 7

If (x_1, y_1, θ_1) are the coordinates and direction of a particle as it leaves a cell, the new coordinates and direction are obtained by first translating the particle by an amount $\Delta x, \Delta y$, and then rotating by an amount $\Delta \theta$.

$$x_2 = (x_1 + \Delta x) \cos \Delta \theta - (y_1 + \Delta y) \sin \Delta \theta$$

$$y_2 = (y_1 + \Delta y) \cos \Delta \theta + (x_1 + \Delta x) \sin \Delta \theta$$

$$\theta_2 = \theta_1 + \Delta \theta$$

Remember that the coordinates and direction are with respect to the coordinate system of the cell that contains the particle.

Values for $\Delta x, \Delta y, \Delta \theta$ are specified for each face. However, since a cell may exist in several lattice positions, the $\Delta x, \Delta y, \Delta \theta$ depend on what lattice position the particle is in and must be given for each possible lattice position. Thus, while we need to describe the geometry of cell type D in Figure 6 only once, we must describe the boundary conditions 6 times, once for each of its lattice positions.

3.2 Hexagonal Lattices

The translation and rotation boundary conditions, $\Delta x, \Delta y, \Delta \theta$ can be computed from simple formulas if we adopt the convention of numbering faces shown in Figure 7. This is a relative number used only for specifying boundary conditions.

If a particle leaves face 1, and the cell adjacent to face 1 has the same alignment (e.g., a uniform lattice), the particle would enter face 4 so that a translation is required, but no rotation. It can be easily verified that if I is the face number of the cell left, J is the face number of the cell entered (adjacent face) and P is the lattice pitch,

$$\Delta x = P \cos [(I-2)\pi/3]$$

$$\Delta y = P \sin [(5-I)\pi/3]$$

$$\Delta \theta = (J-I-3)\pi/3$$

As an example of boundary conditions for a hexagonal lattice, consider the 6/1 lattice shown in Figures 6 and 8. There are several steps.

1. Draw the supercell and one ring of adjacent supercells identifying each cell type.
2. Assign lattice positions to the cells in the supercell and use the same scheme for the adjacent supercells. (The lattice position is the number inside the C or D in Figure 8.).
3. Identify the cells to be explicitly represented (we have chosen C and the D in lattice position 2, D_2) and write in the face numbers following the convention of Figure 7.
4. Write in the face numbers for the remaining cells of the supercell rotating and translating the cell so that it is in the same position as the cell explicitly represented. Thus, face 2 in D_2 faces C_1 , so that face 2 in D_3 , D_4 , D_5 , D_6 , and D_7 must also face C_1 . Having filled in face 2, the remaining faces are filled in proceeding counter-clockwise.
5. Copy the face numbers onto the adjacent supercells.
6. Form the table in Figure 8. For example, face 1, lattice position 1 is adjacent to face 2, lattice position 4.

Figure 7

Lattice Position	Face	Adjacent Lattice Position						Face	Adjacent Face Number					
		1	2	3	4	5	6		1	2	3	4	5	6
1		4	5	6	7	2	3		2	2	2	2	2	2
2		3	1	7	5	4	6		3	5	1	4	6	5
3		4	1	2	6	5	7		3	6	1	4	6	5
4		5	1	3	7	6	2		3	1	1	4	6	5
5		6	1	4	2	7	3		3	2	1	4	6	5
6		7	1	5	3	2	4		3	3	1	4	6	5
7		2	1	6	4	3	5		3	4	1	4	6	5

Figure 8

3.3 Square Lattices

The boundary constants, Δx , Δy , $\Delta \theta$ can be computed from simple formulas if we adopt the convention of numbering faces shown in Figure 9. This is a relative number used only for specifying boundary conditions.

If I is the face number of the cell left, J is the face number of the cell entered (adjacent face), and P is the lattice pitch, the formulas are:

$$\Delta x = P \cos [(I-2)\pi/2]$$

$$\Delta y = P \sin [(I-2)\pi/2]$$

$$\Delta \theta = (J-I-2)\pi/2$$

Figure 9

4. Input Preparation Facilities

It would be very difficult for the user to input the geometrical description of a lattice in terms of vertices, sides, etc. Programs have been written which accept geometrical descriptions in a user-oriented form and convert it to the detailed form. These programs are discussed in detail in the documentation of the ASP and LAGER modules. Only their features are discussed here to summarize the input preparation facilities available.

4.1 ASSEMBLY Data Set

Four basic sub-assembly geometries have been defined for inclusion in the ASSEMBLY data set; cluster, annular regions with ribs, square regions, and hexagonal regions. The data required for each is concise and in user terms. These descriptions could be taken directly from engineering drawings. The user describes as many sub-assemblies as are required to form a complete cell. For example, a cell can be made up of an annular sub-assembly surrounded by a hexagonal sub-assembly, or a cluster surrounded by an annular surrounded by a hexagonal sub-assembly. Each of the sub-assemblies are defined in separate records of the ASSEMBLY data set and catalogued under the cell name. The user need only insure that the sub-assembly boundaries are consistent.

4.2 Execution-Time Modifications

The user may wish to modify the physical descriptions on the ASSEMBLY data set to obtain the mathematical description used in the calculations. The ASP module will accept simple instructions from the user which -

1. Use the rib data to modify annular region sizes.
2. Homogenize adjacent regions into one region.
3. Subdivide an annular region into several annular sub-regions.
4. Subdivide a hexagonal region into several hexagonal sub-regions.
5. Subdivide annular hexagonal, or square sub-assemblies into azimuthal sectors.
6. Subdivide the coolant region of a cluster.

Following these modifications, the sub-assemblies fall into one of the following kinds designated by NSAK.

<u>NSAK</u>	<u>Sub-Assembly Kind</u>
2	Annular, no azimuthal subdivision.
3	Annular with azimuthal subdivision.
4	Hexagonal with a central hole and no azimuthal subdivision.
5	Hexagonal with a central hole and azimuthal subdivision.
6	Cluster.
7	Not used.
8	Square with a central hole and no azimuthal subdivision.
9	Square with a central hole and azimuthal subdivision.

4.3 Lattice Patterns

Since boundary condition data (see Figure 8) are complicated, and since there are only a few lattice patterns of interest, the boundary condition data are catalogued on the LATTICE Data Set under a lattice pattern name. To specify a lattice the user need only supply the pattern name and the names of the cells on the ASSEMBLY data set which are to fill the pattern.

JOSHUA DPSTM-500	Title Part B - Data Management	Vol. IV	Page B.O.1
		Rev.	Date 6/70

Contents

B.1 Data Management Techniques

B.2 Lattice Intermediate Data Sets

JOSHUA
DPSTM-500

Title Part B - Data Management

Vol.
IV

Page
B.O.2

Rev.

Date
6/70

(Blank)

Work Done By

W. H. Hightower
H. C. Honeck

Report Written By

H. C. Honeck

Contents

	<u>Page</u>
1. Introduction	B.1.3
2. Factors Influencing Data Management	B.1.4
3. Core Storage Allocation	B.1.8
4. Data Block Management - MANAGE	B.1.12
5. Zero Elimination	B.1.17
6. Pointer Variables	B.1.20
7. Listing of MANAGE	B.1.23 thru B.1.31

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.2
		Rev.	Date 2/70

(Blank)

1. Introduction

The feasibility of developing the lattice physics subsystem rests entirely on our ability to efficiently handle the large amount of data. To illustrate this point, consider some typical sizes of the matrices required. At least 50, and perhaps 100 regions are needed to describe the complex two-dimensional geometry. While 50 energy groups might be adequate for many studies, 100 groups would be required for some studies. Thus, if we take the design requirement of 100 groups and 100 regions, the transport kernel would have 10^6 elements. If we only keep the down-scattering half of the scattering matrices for 20₄ materials, 10^5 elements are needed. The flux alone has 10^4 elements. Adding cross sections and other miscellaneous storage indicates that some 2×10^6 data are required.

The modest computational facilities at SRL (IBM 360/MP65) has a 10^6 byte (250K word) fast core storage. This computer must be operated in a multiprogramming environment (MVT) to obtain the efficiency necessary to handle the large workload. The operating system takes nearly 250K bytes leaving 750K bytes to be shared by several jobs. A 240K byte (60K word) module core size was chosen since this appears to be consistent with the demands of MVT and will permit the code to be run on a 64K word CDC 6600 or UNIVAC 1108. If we assume that 30K words will be used by programs and data I/O buffers, about 30K words are available to hold the actual data. We must then achieve a data reduction from 2×10^6 to 3×10^4 (a factor of 66). In doing this we must not unduly increase the modules' running time since they will be major production modules at SRL and will be run many times a day.

Several methods are used to reduce the volume of data that must be in core at any one time.

- physical approximations
- partitioning into modules
- dense packing of data
- shuffling data between core and disk

The major physical approximation is made in the treatment of the transport matrices by subdividing the space into zones coupled together by currents. This has the effect of transforming the transport matrix into small submatrices along the matrix diagonal. This approximation will be discussed later in the section on transport matrices. Partitioning into modules is discussed in Section C.1, the main effect on storage being the reduced program size. The last two methods, dense packing and data shuffling, are discussed in this section.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.4
		Rev.	Date 2/70

2. Factors Influencing Data Management

Several factors must be discussed to show why the in-core data management scheme used was chosen. These factors are:

- associations
- zero elimination
- flexibility

The association between data in the lattice physics subsystem is quite complex and is best illustrated by the geometry and material tables. This is shown in Figure 1. A supercell is divided into cells, a cell into sub-assemblies, a sub-assembly into regions, a region into sub-regions, a sub-region into sides, and sides into vertices and radii. Materials are associated with regions and are divided into isotopes which are associated with cross section data. One or more sub-assemblies constitutes a zone coupled to other zones through boundary conditions on the faces. The quantity of actual data (vertex coordinates, radii, volumes, areas, concentrations) is small but the data needed to express the relationships can be quite large. The arrows in Figure 1 have a head on both ends implying that the associations must be made in both directions. One part of the calculation may need to know what sub-regions are in a region. Another part may need to know in which region a given sub-region is located.

Figure 1. Association Between Geometric Parts and Material Parts.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.6
		Rev.	Date 2/70

The elimination of zeros is best illustrated by the scattering kernels (see Figure 2). In the thermal energy region there are upscattering contributions (shaded area above the diagonal in Figure 2) but, since they can be computed by detailed balance from the downscattering terms, there is no need to store them. The size of the non-zero downscattering portion depends on the isotope. For hydrogen all terms are non-zero. For deuterium about 2/3 of the terms are non-zero. For materials heavier than oxygen the shaded part is very small except for inelastic terms at high energies. When isotopes are collected into materials the kernel size is dictated by the lightest isotope in the material. A typical lattice might use 20 materials and 10 isotopes. If the data is kept by isotope (and materials formed only when needed) very few dense matrices and many sparse matrices are needed. It is then very economical to store only the non-zero data even though additional data must be stored to indicate what data is non-zero.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.7
		Rev.	Date 2/70

Figure 2. Illustration of the Non-Zero Data (Shaded) in a Scattering Kernel.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.8
		Rev.	Date 2/70

Flexibility is necessary so that storage allocation can be determined at execution time without recompilation and so that storage areas can be reallocated during execution. For example, the flux is stored by sub-region and energy group. There are no zeros in the array and a simple two-dimensional Fortran array is quite adequate. In some problems we want to increase the number of sub-regions at the expense of using fewer groups, or vice versa, so that we do not want to allocate a flux array with fixed dimensions. Variable dimensioning would work well for this example, but SRL has had difficulties using this feature of the manufacturer-supplied compiler. Instead, most dense multi-dimensional arrays have been treated as one-dimensional arrays with the appropriate indexing computed in the Fortran program. Several programming tricks are used to reduce the computational overhead.

3. Core Storage Allocation

All data is stored in named `COMMON` areas (see Figure 3). There are several small areas for group structures and control data. The `/SCRACH/` area is used by subroutines as a scratch area and each subroutine structures the area as needed. The `/TEMP/` area is used to transmit data between subroutines of a module. Each module uses a different `/TEMP/` structure. These structures will be discussed in later sections. We will be concerned here with the `/DATA/` area.

The size of the `/DATA/` area is specified in the main program of each module with the statements,

```
COMMON/DATA/DATA(20000)
KXX=20000
```

KXX is used in the programs to test for overflow of the DATA array. All other routines in the module use the statement,

```
COMMON/DATA/DATA(1)
```

so that only two statements need be changed to allocate more or less space.

Figure 3. Schematic layout of named COMMON and an example of some blocks packed into the DATA array and stored on the disk.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.10
		Rev.	Date 2/70

Many separate blocks of data are packed into the DATA array. For example, the blocks called NAME, CØMP, GEØM, FLUX, XSEC, and SMAT are shown packed into the DATA array in Figure 3. We use the symbol K as an index to the DATA array. Since DATA is a REAL*4 variable, K counts 4 bytes groups. Each block in the DATA array has an origin which locates the block in the array. These origins are written as K ØRG, where the blank is replaced with one character designating the block. For example, KCØRG (origin of the CØMP block), KFØRG (origin of the FLUX block), etc. The block origin is the index of the last word of the preceding block rather than the first word of the block so that we do not have to continually subtract one from the origin. Thus, the Nth word of the FLUX block is DATA(N+KFØRG), not DATA(N+KFØRG-1).

A program written in this way would be very difficult to read (and hence debug) since nearly every statement would contain the symbol DATA with an index. In an effort to provide more mnemonic content to the symbols, we give the DATA array different names. When we refer to the CØMP block we use the array name CØMP, references to the FLUX block are through the array name FLUX, etc. Thus, the Nth word of the CØMP block is CØMP(N+KCØRG), the Nth word of the FLUX block is FLUX(N+KFØRG), etc. The start of a routine then contains the statements

```

COMMON/DATA/DATA(1)
REAL*4 CØMP(1),FLUX(1), etc.
EQUIVALENCE(DATA(1),CØMP(1),FLUX(1),etc.)

```

so that all of these arrays are the same as the DATA array but with different names.

Frequently we need to pack integers and pointers into these arrays. Since all the integers needed are less than 32K we need only 2 bytes. Integer arrays are defined by INTEGER*2 K PT(2,1) where the blank is replaced by the character designating the block of interest, e.g., KFPT, KCPT, etc. These integer arrays are also equivalenced to the DATA array. This is illustrated in Figure 4 for the composition block CØMP. This method allows us to reference the left 2 bytes of CØMP(K) by KCPT(1,K), and the right 2 bytes by KCPT(2,K).

All data in the DATA array is represented as either a single precision, 4-byte, floating point number or a 2-byte integer except for character data (names). All names (of cells, regions, materials, isotopes, etc.) are 8-character names and are packed in the NAME block which is always the first block in the DATA array.

If a pointer (e.g., KRGNAM for pointer K to the region name) occurs in the GEØM block, the 8-character name is in DATA(KRGNAM) and DATA(KRGNAM+1).

Figure 4. Illustration of Splitting a 4-byte Word into Two 2-byte Integers.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.12
		Rev.	Date 2/70

4. Data Block Management

Before describing the internal structure of the data blocks, we will discuss several facilities provided to manipulate entire blocks. As shown in Figure 3 the data blocks are also stored on the disks. The blocks stored on disk are the interfaces between modules. The JOSHUA data manager is used to do the actual reading and writing of blocks so that each block must have a unique JOSHUA name.

The general form of a block name is

A1.'LIDS'.A2.A3.BLØK.N1.N2.N3

where:

A1 - temporary data set name 'TEMP' or 'SCRATCH'
A2 - 8-character lattice name
A3 - 8-character lattice version name
BLØK - 4-character block type name
N1 - case number (used for internal searches)
N2 - integer (usually the energy band number)
N3 - integer (usually the zone number).

For example,

'TEMP'.'LIDS'.'MK-14'.'VERG'.'TMAT'.1.2.5

refers to the TMAT (transport matrix) data for energy band number 2, zone number 5, for lattice MK-14, version VERG, Case 1.

All blocks are read or written with the same list structure; an integer length of the block, and the data in the block. If NX is the length of the block, and KØRG is its origin, the list is

NX,(DATA(N+KØRG),N=1,NX) .

At execution time all blocks read by a module would have the same first three qualifiers, A1, A2, A3, so that within the module only the last four qualifiers (BLØK.N1.N2.N3) are used to refer to a block. Thus, in the last example, in the program we would refer to 'TMAT'.1.2.5 for the TMAT data for case 1, energy band 2, zone 5.

A block manager subroutine, `MANAGE`, was written to aid the programmer in manipulating these data blocks*. `MANAGE` has several entry points, each of which performs a management function on the `DATA` array. These functions are described below. The symbols are those used in the `MANAGE` subroutine.

<u>Argument Symbol</u>	<u>Internal Symbol</u>	<u>Type</u>	<u>Description</u>
A1P	A1	R*8	Data set name.
A2P	A2	R*8	Lattice name.
A3P	A3	R*8	Lattice version.
BLØKP	BLØK(N)	R*4	Record type name.
N1P	N1(N)	I*4	Case number.
N2P	N2(N)	I*4	Energy band number (usually).
N3P	N3(N)	I*4	Zone number (usually).
KØRG	KØRG(N)	I*4	Block origin.
LDBP	LDB(N)	I*4	Length of block.
KXP	KXX	I*4	Length of <code>DATA</code> array.
KP		I*4	Last location of <code>DATA</code> array used by calling routine.
	NXX	I*4	Maximum number of blocks (=100).
	NX	I*4	Current number of blocks in catalogue.
	KX	I*4	Last location of <code>DATA</code> array used.

Data are passed between the various subroutines of `MANAGE` through the `CØMMØN/DATCAT/` area:

```
CØMMØN/DATCAT/A1,A2,A3,KX,KXX,NX,NXX,BLØK(100),
              N1(100),N2(100),N3(100),KØRG(100),LDB(100) .
```

- * Since nearly all `JOSHUA` reads and writes are done in the subroutine, and since the number of blocks involved is small (~20), this routine can easily be rewritten for use at another installation which uses sequential disk access methods.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.14
		Rev.	Date 2/70

The following paragraphs list the entry point name and arguments. The second set of arguments enclosed in parentheses indicates whether the arguments are input (I) to, or output (Ø) from MANAGE.

CLEAR(A1P,A2P,A3P,KXP) (I,I,I,I)

The arguments are stored and the DATA array and catalogues are cleared.

ØPENDB(BLØKP,N1P,N2P,N3P,KØRGP,KXP) (I,I,I,I,Ø,Ø)

If a block with this name is already catalogued, it is deleted and the arrays packed. The block is assigned following the last block catalogued.

CLØSDB(BLØKP,N1P,N2P,N3P,KP) (I,I,I,I,I)

If this is the block last opened, and the length does not exceed the size of the DATA array, the block is closed and the length entered into the catalogues. If this block is not the one last opened, or if no data was entered into the DATA array, the block is deleted from the catalogues.

DELEDB(BLØKP,N1P,N2P,N3P) (I,I,I,I)

All blocks with this name are deleted and the DATA array and catalogues are packed. If N1P,N2P, or N3P is negative, it is interpreted as "all". For example, 'TMAT',-1,-1,-1 deletes all transport matrices.

PACK

All unused space is removed from the DATA array and catalogues.

REØPDB(BLØKP,N1P,N2P,N3P,KØRGP,KP,KXP) (I,I,I,I,Ø,Ø,Ø)

If the block is found, it is moved to the end and the DATA array packed. The block is reopened and the status (KØRGP,KP,KXP) passed to the calling program. If the block is not found, the block is opened and KP=KØRGP.

CLERDB(BLØKP,N1P,N2P,N3P) (I,I,I,I)

If block is found, it is cleared. If block is not found, a block is opened and cleared.

READDB(BLØKP,N1P,N2P,N3P,KØRGP,*) (I,I,I,I,Ø,Ø)

If block is already in the catalogue, it is deleted and the data packed. The block is read from the disk and placed following the last block already catalogued. If the block cannot be found on disk, the error return (*) occurs. Note that all blocks catalogued must be closed. If a READDB is issued after an ØPENDB and before a CLØSDB, the open block will be overwritten by the block read from disk.

WRITDB(BLØKP,N1P,N2P,N3P) (I,I,I,I)

If the block is found, it is written on the disk.

FINDDB(BLØKP,N1P,N2P,N3P,KØRGP,*) (I,I,I,I,Ø,Ø)

If the block is found, the origin is passed to the calling routine. Otherwise, the error return (*) occurs.

LISTDB

Print the catalogues on the system output file.

ALØCDB(BLØKP,N1P,N2P,N3P,KØRGP,LDBP) (I,I,I,I,Ø,I)

If the block is not found, the requested space is allocated (combination of ØPENDB and CLØSDB). If the block is already catalogued and the space requested is less than previously allocated, the old space is cleared and used. If the space requested is greater than previously allocated, the old space is deleted, the arrays packed, and new space allocated.

RENAME(BLØKP,N1P,N2P,N3P,BLØKN,N1N,N2N,N3N) (I,I,I,I,I,I,I,I)

The block name BLØKP,N1P,N2P,N3P in the catalogue is replaced with the new name BLØKN,N1N,N2N,N3N.

CØUNT('module name') (I)

The MANAGE routine will keep track of the maximum location in the DATA actually used during a module execution. A call to the COUNT entry point will cause this value to be printed along with the 8-character module name.

Two points should be made in connection with the block manager. First, it is not intended to be a general-purpose data manager.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.16
		Rev.	Date 2/70

It was developed specifically for the type of data blocks used in the lattice physics sub-system. The second point is that the programmer must take care to keep up with his block origins, since the block manager packs the DATA array whenever a DELEDB or REØPDB is issued causing the block origins to be changed.

5. Zero Elimination

The use of equivalent floating point and integer arrays allows us to imbed control integers in the data array. For example, if the scattering matrix for an isotope, $\sigma(I \rightarrow J)$, is non-zero for $IA(J) \leq I \leq IB(J)$, we might list the essential data for final group J as:

$$J, IA, IB, \sigma(J), \sigma(IA \rightarrow J), \sigma(IA+1 \rightarrow J), \dots, \sigma(IB \rightarrow J)$$

where $\sigma(J)$ is the cross section obtained by summing the elements over all J . These data are placed in the floating point array $SMAT(K)$ and integer array $KSPT(N,K)$ starting at some index KJ as shown in Figure 5.

REAL*4 INTEGER*2
 SMAT(K) = KSPT(N,K), N=1,2

Figure 5. Example of Storing Scattering Matrix Data.

A sample Fortran program to access these data is:

```
J=KSPT(1,KJ)
IA=KSPT(1,KJ+1)
IB=KSPT(2,KJ+1)
SIG(J)=SMAT(KJ+2)
K=KJ+3
DØ 10 I=IA,IB
S(I,J)=SMAT(K)
10 K=K+1
```

There are, of course, several ways in which the Fortran could be written. For example,

```
DØ 10 I=IA,IB
10 S(I,J)=SMAT(KJ+I - IA+3)
```

works just as well. Several methods are used in RAHAB, the choice being dictated by compiler characteristics, complexity of the loop under consideration, and (mainly) programmer preference.

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.20
		Rev.	Date 2/70

6. Pointer Variables

In the previous example the data can be accessed if we can locate where the data starts, that is, the value of KJ. Let us assume that an index KJ is known for the first J(=1). The variable KNXTJ in Figure 5 is called the pointer variable and is the index (relative to the block origin) where the data for the next J starts.

Four conventions have been followed in using pointer variables:

- The pointer variable is always relative to the block origin so that relocating the block in the DATA array does not change the pointer variable values.
- A pointer variable in one block (A) pointing to data in another block (B) is relative to the origin of the second block (B).
- The pointer variable always appears in the right-most two bytes of a word.
- A pointer variable with a value of zero signals the last entry.

As an example, suppose that we wish to locate the scattering matrix data for J=5 and we know that the data for J=1 starts at KJ=2+KSØRG. The Fortran program might be

```

      KJ=2+KSØRG
10  IF(KSPT(1,KJ).EQ.5) GØ TØ 20
      KJ=KSPT(2,KJ)+KSØRG
      IF(KJ.GT.KSØRG) GO TO 10
      (error, J=5 not given)
20  (J=5 data starts at current value of KJ)

```

A better example of the use of pointer variables is the geometry tables where association information constitutes the bulk of data. A region description is shown in Figure 6. This region is a part of sub-assembly NSA which is described starting at K=KSA+KGØRG. (The symbols GEØT(K), KGPT(N,K), and KGØRG are used to refer to the geometry tables.). The region is divided into ISRX sub-regions whose numbers are NSR and which are described starting at K=KSR+KGØRG. The region has an 8-character name stored in the NAME block at location K=KRGNAM+0. (The NAME block is always first in the DATA array

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.21
		Rev.	Date 2/70

and hence has a zero origin).

The pointer variable is used four different ways in this example. First, it points to the next element of this type (next region). If needed, another pointer could point to the preceding element of this type. Second, it points down to sub-elements (sub-regions) of this element. Third, it points up to the element (sub-assembly) of which this element (region) is a sub-element. Fourth, it points across to related data (region name). Materials are also related to regions but, for convenience, the material pointers are contained in the sub-region descriptions.

A major advantage of storing data in this manner is that it does not matter where or in what order these sub-blocks (such as the one in Figure 6) are stored in the block. The sub-blocks can then be densely stored in the order encountered in the input description of the geometry. Each type of sub-block forms a chain. There is a chain for cells, sub-assemblies, regions, sub-regions, sides, faces, vertices, and radii within the geometry tables. These chains are linked with pointers, cross-linked with pointers, and linked to chains in other blocks with pointers.

Figure 6. Example of data describing a region.

C	SUBROUTINE MANAGE	
C		
C	ENTRY POINTS/ARGUMENTS	EXPLANATION
C	-----	-----
C	CLEAR	CLEAR DATA ARRAY AND CATALOGUES
C	(A1, A2, A3, KXP)	
C		
C	OPENDB	OPEN A BLOCK AT END OF DATA ARRAY
C	(BLOK, N1, N2, N3, KOR G, KXP)	
C		
C	CLOSDB	CLOSE A BLOCK AND ISSUE A 'J WRITE'
C	(BLOK, N1, N2, N3, KP)	
C		
C	DELEDB	DELETE A BLOCK (DONT PACK)
C	(BLOK, N2, N2, N3)	

```

C      PACK                                PACK DATA ARRAY
C      ( )
C
C      REOPDB                                REOPEN BLOCK (MOVE TO END OF ARRAY
C      (BLOK,N1,N2,N3,KORG,KP,KXP)        AND PACK)
C
C      CLERDB                                CLEAR A BLOCK
C      (BLOK,N1,N2,N3)
C
C      READDB                                READ A BLOCK ('J READ'),PLACE AT END
C      (BLOK,N1,N2,N3,KORG,&SN)
C
C      WRITDB                                WRITE A BLOCK ('J WRITE')
C      (BLOK,N1,N2,N3)
C
C      FINDDB                                FIND BLOCK ORIGIN
C      (BLOK,N1,N2,N3,KORG,&SN)
C
C      DUMPDB                                DUMP BLOCK USING 'CHECK'
C      (BLOK,N1,N2,N3)
C
C      LISTDB                                PRINT CATALOGUES
C      ( )
C
C      ERROR
C      (
C
C      ALOCDB                                OPEN AND LOOSE A BLOCK OF LENGTH LDB
C      (BLOK,N1,N2,N3,KORG,LDB)
C
C      COUNT                                COUNT MAX NO OF ENTRIES IN DATA ARRAY
C      (MODULE)

```

C KX POINTS TO LAST FILLED POSITION

C.....
C ALL WRITES AND READS WILL HAVE A TEMPLATE CONSISTING OF SEVEN NAME

NO.	NAME	DEFINITION	
1	A1	ANAME	(REAL *8)
2	A2	ANAME	(REAL * 8)
3	A3	ANAME	(REAL * 8)
4	BLOK	RECORD TYPE NAME	(REAL * 4) (BLOK)
5	NCASE	CASE	(INTEGER *4) (N1)
6	NB	ENERGY BAND NUMBER	(INTEGER *4) (N2)
6	NBS	BUCKLING NUMBER	(INTEGER *4) (N2)
7	NZN	ZONE NUMBER	(INTEGER *4) (N3)

C

```
COMMON /DATA/ DATA(1)
REAL *8 A1,A2,A3,A1P,A2P,A3P
COMMON/DATCAT/A1,A2,A3,KX,KXX,NX,NXX,BLOK(100),N1(100),N2(100),
X N3(100),KORG(100),LDB(100)
REAL *8 BLOK8/'/' ,MODULE
EQUIVALENCE (BLOK8,BLOKP)
REAL *4 BLANK/'/' ,RTYPE(16)
DATA RTYPE/'NAME','CARD','COMP','GEOT','ZONE','XSEC','TMAT','SMAT'
X,'RESC','FLOX','JIN','JOUT','HOXS','HORI','HOSM','HOFL'/'
J A1,A2,A3 -> V
```

C.....

```
ENTRY CLEAR (A1P,A2P,A3P,KXP)
KXX=KXP
A1=A1P
A2=A2P
A3=A3P
NXX=100
NX=0
KX=0
```

10

```
NCOUNT=0
DO 10 I=1,KXX
DATA(1)=0.0
DO 20 I=1,NXX
BLOK(I)=BLANK
N1(I)=0
N2(I)=0
N3(I)=0
KORG(I)=0
20 LDB(I)=0
RETURN
```

C.....

```
ENTRY OPENDB(BLOKP,N1P,N2P,N3P,KORGP,KXP)
KORGP=KX
IF(NX.EQ.0) GO TO 30
CALL DELETE (BLOKP,N1P,N2P,N3P,0)
30 CALL ASGNCT(BLOKP,N1P,N2P,N3P,KORGP)
KXP =KXX
RETURN
```

C.....

```
ENTRY CLOSOB(BLOKP,N1P,N2P,N3P,KP)
NX=NX+1
IF(BLOK(NX).EQ.BLANK.AND.N1(NX).EQ.N1P.AND.N2(NX).EQ.N2P.AND.N3(NX)
X).EQ.N3P) LDB(NX)=KP-KORG(NX)
IF (KP.GT.KXX) GO TO 900
IF (LDB(NX).GE.0) GO TO 35
BLOK(NX)=BLANK
LDB(NX)=C
NX=NX-1
RETURN
95 KX=KP
IF(NCOUNT.LT.KX) NCOUNT=KX
RETURN
```

C.....

```
ENTRY DELEDB(BLOKP,N1P,N2P,N3P)
IF(NX.LE.0) RETURN
```

JOSHUA DPSTM-500	Title Data Management Techniques	Vol. IV	Page B.1.26
		Rev.	Date 2/70

```

      CALL DELETE (BLOKP,N1P,N2P,N3P,0)
      RETURN
C.....
      ENTRY PACK
      CALL PACKP
      RETURN
C.....
      ENTRY REOPDB(BLOKP,N1P,N2P,N3P,KORGP,KP,KXP )
      KOGP=KX
      CALL FIDNAM (BLOKP,N1P,N2P,N3P,IX)
      IF (IX.GT.NX) GO TO 40
      CALL MOVE (KORG(IX),LDB(IX),KOGP )
      CALL DELETE (BLOKP,N1P,N2P,N3P,IX)
40     CALL ASGNCT(BLOKP,N1P,N2P,N3P,KOGP)
      NX=NX+1
      LDB(NX)=LDB(IX)
      CALL PACKP
      KP=KORG(NX)+LDB(NX)
      KORGP=KORG(NX)
      KX=KORG(NX)+LDB(NX)
      KXP=KXX
      NX=NX-1
      RETURN
C.....
      ENTRY CLERDB(BLOKP,N1P,N2P,N3P)
      CALL FIDNAM (BLOKP,N1P,N2P,N3P,IX)
      IF(IX.LE.NX) GO TO 50
      CALL ASGNCT (BLOKP,N1P,N2P,N3P,KX)
      IL=KORG(NX+1) + 1
      IH=KXX
      GO TO 52
50     IL= KORG(IX)+1
      IH= KORG(IX)+LDB(IX)
52     DO 51 I=IL,IH
51     DATA(I)=0.0
      RETURN
C.....
      ENTRY READDB(BLOKP,N1P,N2P,N3P,KORGP,*)
      CALL DELETE (BLOKP,N1P,N2P,N3P,0)
      KORGP=KX
J      READ(V.BLOKB,N1P,N2P,N3P,ERR=63) LDBP,(DATA(KORGP+1),I=1,LDBP)
      CALL ASGNCT (BLOKP,N1P,N2P,N3P,KORGP)
      NX=NX+1
      LDB(NX)=LDBP
      KX=KX+LDBP
      IF(KX.GT.KXX) GO TO 900
      IF(NCOUNT.LT.KX) NCOUNT=KX
      RETURN
63     KORGP=-1
      RETURN 1
C.....
      ENTRY WRITDB (BLOKP,N1P,N2P,N3P)
      CALL FIDNAM (BLOKP,N1P,N2P,N3P,IX)
      IF (IX.GT.NX) GO TO 73
      LDBP=LDB(IX)
      KORGP=KORG(IX)

```

```

      IF(LDBP.LE.0) GO TO 73
J      WRITE(V,BLOKB,N1P,N2P,N3P) LDBP,(DATA(KORGP+1),I=1,LDBP)
73     RETURN
C.....
      ENTRY FINLDB(BLOKP,N1P,N2P,N3P,KORGP,*)
      CALL FIDNAM (BLOKP,N1P,N2P,N3P,IX)
      KORGP=-1
      IF (IX.GT.NX) RETURN 1
      KORGP=KORG(IX)
      RETURN
C.....
      ENTRY DUMPLB (BLOKP,N1P,N2P,N3P)
      CALL FIDNAM( BLOKP,N1P,N2P,N3P,IX)
      IF(IX.GT.NX) RETURN
      LDBP=LDB(IX)+ KORG(IX)
      IF (LDBP.LE.KORG(IX)) RETURN
      IF (BLOKP.EQ.RTYPE(1)) GO TO 75
      CALL CHECK(BLOKB,KORG(IX),LDBP)
      RETURN
75     LDBP=LDB(IX)+KORG(IX)
      LBB=KORG(IX)+1
      WRITE (6,76) (1,DATA(1),DATA(I+1),I=LBB,LDBP,2)
76     FORMAT ('1 NAMES ARRAY '/ ' ',100(' ')/(5(2X,12,2X,2A4)))
      RETURN
C.....
      ENTRY ALOCDB( BLOKP,N1P,N2P,N3P,KORGP,LDBP)
      CALL FIDNAM ( BLOKP,N1P,N2P,N3P,IX)
      IF(IX.LE.NX) GO TO 101
      IF(KX+LDBP.GT.KXX)CALL PACKP
99     IF(KX+LDBP.GT.KXX) GO TO 900
      CALL ASGNCT(BLOKP,N1P,N2P,N3P,<X)
      NX=NX+1
      LDB(NX)=LDBP
      KORGP=KX
      KX=KX+LDBP
      IF(NCOUNT.LT.KX) NCOUNT=KX
      RETURN
101    IF (LDB(IX).GE.LDBP) GO TO 102
      CALL DELETD (BLOKP,N1P,N2P,N3P,IX)
      CALL PACKP
      GO TO 99
102    LDB(IX)=LDBP
      KORGP=KORG(IX)
      IL=KORGP+1
      IH=LDBP+KORGP
      DO 103 I=IL,IH
103    DATA(I)=0.0
      RETURN
C.....
      ENTRY COUNT(MODULE)
      WRITE(6,110) MODULE,NCOUNT
110   FORMAT(/'0 IN ',A8,' THE MAXIMUM NUMBER OF ENTRIES IN THE DATA AREA
      XY IS ',I5,'.')
      RETURN
      ENTRY RENAMF(BLOKP,N1P,N2P,N3P,BLDKPP,N1PP,N2PP,N3PP)
      CALL FIDNAM(BLOKP,N1P,N2P,N3P,IX)

```

```

      IF (IX.GT.NX) RETURN
      BLOK (IX)=BLOKPP
      N1 (IX)=N1PP
      N2 (IX)=N2PP
      N3 (IX)=N3PP
      RETURN
J 900 STOP 'MANAGE, DATA ARRAY OVERFLOW'
      END
      SUBROUTINE LISTDB
C.... THIS SUBROUTINE BELONGS TO DATA MANAGER.  LIST DATA BLOCK
C
      REAL *8 A1,A2,A3,A1P,A2P,A3P
      COMMON/DATCAT/A1,A2,A3,KX,KXX,NX,NXX,BLOK(100),N1(100),N2(100),
X          N3(100),KORG(100),LDB(100)
      IF (NX.LE.0) RETURN
      WRITE(6,90) A1,A2,A3
90    FORMAT(1H1,5X, 'A LIST OF CATALOGUES'/
X          6X, '-----'/6X,A8, ' ',A8,' ',A8,'
X -> V'/6X,'RECORD NAMES ',5X,'ORIGIN',5X,'LENGTH'/
X5X,25(' '),5X,6(' '),5X,6(' '))
      WRITE(6,91)(BLANK,BLOK(I),N1(I),N2(I),N3(I),KORG(I),LDB(I),I=1,NX)
91    FORMAT(5(5X,A1,'V.',A4,2(' '),I4),5X,I6,5X,I6/)
      RETURN
      END
      SUBROUTINE DELETE (BLOKPP,N1P,N2P,N3P,ITEST)
C THIS ROUTINE DELETES THE NAMES FROM THE CATALOGUES BY PLACING
C 999 IN PLACE OF BLOK(I). IF ITEST EQ.0 NAME MUST BE FOUND FIRST
      REAL *8 A1,A2,A3
      COMMON/DATCAT/A1,A2,A3,KX,KXX,NX,NXX,BLOK(100),N1(100),N2(100),
X          N3(100),KORG(100),LDB(100)
      IF (NX.EQ.0) RETURN
      IT=ITEST
      IF (ITEST.EQ.0) GO TO 1
      IF (IT.GT.VX) RETURN
      BLOK(IT)=999.0
      RETURN
01    DO 10 N=1,NX
      IF (BLOKPP.NE.BLOK(N)) GO TO 10
      IF ((N1P.NE.N1(N)).AND.(N1P.GE.0)) GO TO 10
      IF ((N2P.NE.N2(N)).AND.(N2P.GE.0)) GO TO 10
      IF ((N3P.NE.N3(N)).AND.(N3P.GE.0)) GO TO 10
      BLOK(N)=999.0
10    CONTINUE
      CALL PACKP
      RETURN
      END
      SUBROUTINE PACKP
C THIS ROUTINE PACKS BLANK SECTIONS OUT OF THE DATA ARRAY AND UPDATE
C S THE CATALOGUES AS NECESSARY
      REAL *8 A1,A2,A3
      COMMON/DATCAT/A1,A2,A3,KX,KXX,NX,NXX,BLOK(100),N1(100),N2(100),
X          N3(100),KORG(100),LDB(100)
      COMMON/DATA/ DATA(1)
      IF (NX.LE.0) RETURN
      INX=0
      ILORG=0

```

```

LDBL=0
DO 10 I=1,NX
  IF (LDB(I).LT.0) LDB(I)=MAX(9,KORG(I+1)-KORG(I))
  IF (BLOK(I).EQ.999.0) GO TO 10
  CALL MOVE (KORG(I),LDB(I),LDBL)
  INX=INX+1
  LDB(INX)=LDB(I)
  BLOK(INX)=BLOK(I)
  N1(INX)=N1(I)
  N2(INX)=N2(I)
  N3(INX)=N3(I)
  KORG(INX)=LDBL
9   LDBL=LDBL+LDB(I)
10  CONTINUE
  NX=INX
  KX=LDBL
  LDBL=LDBL+1
DO 20 I=LDBL,KXX
10  DATA(I)=0.0
  RETURN
END
SUBROUTINE FIDNAM(BLOKP,N1P,N2P,N3P,IX)
C   THIS ROUTINE FINDS THE CATALOGUED NAMES IN THE CATALOGJE
C   IF NAME CANNOT BE FOUND IX IS SET TO NX+1
  REAL *8 A1,A2,A3
  COMMON/DATCAT/A1,A2,A3,KX,KXX,NX,NXX,BLOK(100),N1(100),N2(100),
X   N3(100),KORG(100),LDB(100)
  DO 10 I=1,NX
    IF (BLOKP.EQ.BLOK(I).AND.N1P.EQ.N1(I).AND.N2P.EQ.N2(I).AND.
X   N3P.EQ.N3(I)) GO TO 12
10  CONTINUE
  IX=NX+1
  RETURN
12  IX=I
  RETURN
END
SUBROUTINE ASGNCT(BLOKP,N1P,N2P,N3P,KORGP)
C   THIS ROUTINE SETS UP THE CATALOGUE AT NX+1
  REAL *8 A1,A2,A3
  COMMON/DATCAT/A1,A2,A3,KX,KXX,NX,NXX,BLOK(100),N1(100),N2(100),
X   N3(100),KORG(100),LDB(100)
  IF (NX.GE.NXX) RETURN
  NX=NX+1
  BLOK(NX)=BLOKP
  N1(NX)=N1P
  N2(NX)=N2P
  N3(NX)=N3P
  KORG(NX)=KORGP
  NX=NX-1
  RETURN
END
SUBROUTINE MOVE (IORA,LENGTH,IORB)
C   THIS ROUTINE MOVES LENGTH LOCATIONS FROM IORA TO IORB
C   ASSUMES NO OVERLAP
  COMMON/DATA/DATA(1)
  IF (LENGTH.LE.0.OR. IORA.LT.0 .OR. IORB.LT.0) RETURN

```

```

20 DO 10 I=1,LENGTH
DATA(IORB+I)=DATA(IORA+I)
RETURN
END
SUBROUTINE CHECK (TITLE,ISTART,ISTOP)
C==== THIS ROUTINE WRITES ON NOUT (6), SIX VARIABLES PER ROW BEGINING
C AT ISTART+1 AND ENDING AT ISTOP
COMMON/DATA/DATA(1)
INTEGER*2 KGPT(2,1),KCPT(2,1)
EQUIVALENCE (DATA(1),KGPT(1,1),KCPT(1,1))
COMMON/SCRACH/ EDIT(18),FMT(26)
REAL *8 TITLE
REAL *4 KBL/' ',TEST/0.0/,TES/0.0/
A,SPACE1/'3X,I',SPACE2/'5,2X',FMI1/'',I6,'',FMI2/'16,'',FMR1/'',E12
B'/',FMR2/'0.6', '/',LPP/'(' '/',RPP/'1X)' /,NEG/ZFFFFC000/
INTEGER*4 ITS/0/,ICOL(6),IDIT(18)
INTEGER*2 KACP(2,1)
LOGICAL*1 ED(72), FM(104),ONE/'1',T(4)
EQUIVALENCE (EDIT(1),ED(1)),(DATA(1),KACP(1,1)),(FMT(1),FM(1)),
A(TEST,T(1)), (IDIT(1),EDIT(1))
ISTO=ISTOP
IS=ISTART +1
NOUT=6
LENFMT=26
LENEDT=18
FMT(1)=LPP
DO 10 I=2,22,4
FMT(I)=SPACE1
FMT(I+1)=SPACE2
10 CONTINUE
FMT(26)=RPP
FM(5)=ONE
NPAGE=(ISTO+1-IS)/300
IF (MOD(ISTO+1-IS,300).NE.0) NPAGE=NPAGE+1
ISP=IS
ISTP=ISTO
ISTO=IS+299
IF(ISTO.GT.ISTP) ISTO= ISTP
DO 120 IPASS=1,NPAGE
WRITE (NOUT,5)IS,TITLE,ISTO, IPASS,NPAGE
5 FORMAT ('1',I2('>'),' FROM ',I5,' ',I2('<'),1X,A8,2X,I2('>'),
' TO ',I5,' ',I2('<'),' PAGE ',I4,' OF ',I4,' PAGES')
MODA=MOD(ISTO-IS+1,6)
IDISP=(ISTO+1-IS)/6
DO 40 II=1,6
40 ICOL(II)=IS-1+(II-1)*IDISP
IF(MODA.EQ.0) GO TO43
DO 41 II=1,MODA
III=II+1
DO 42 JIJ=III,6
42 ICOL(JIJ)=ICOL(JIJ)+1
43 CONTINUE
NULOC=6
DO 100 IPDS=IS,ISTO,6
DO 44 II=1,6
ICOL(II)=ICOL(II)+1

```

```

44  CONTINUE
    DO 11 IJ=4,24,4
      FMT(IJ)=KBL
      FMT(IJ+1)=KBL
11  CONTINUE
    DO 12 IJ=1,LENEDT
      EDIT(IJ)=0.0
12  CONTINUE
      LENEDT=0
      IF ( IPOS + 5 .GT. ISTO)  NMULOC=ISTO-IPOS +2
      IFT=0
      DO 90 J=1,NMULOC
        TEST=DATA(ICOL(J))
        ITE=1(1)
C----- INTEGER ?
        IFT=IFT+4
        IF(ITE.GE.217.OR.ITE.LE.29) GO TO 30
C----- REAL
        IDIT(LENEDT+1)=ICOL(J)-ISP+1
        EDIT(LENEDT+2)=TEST
        FMT(IFT)=FMR1
        FMT(IFT+1)=FMR2
        LENEDT=LENEDT+2
        GO TO 90
C----- INTEGER
30  CONTINUE
        IDIT(LENEDT+1)=ICOL(J)-ISP+1
        IF(ITE.GT.217)EDIT(LENEDT+2)=NEG
        ED(4*(LENEDT+2)-1)=T(1)
        ED(4*(LENEDT+2))=T(2)
        ITE=T(3)
        IF(ITE.GT.217) EDIT(LENEDT+3)=NEG
        ED(4*(LENEDT+3)-1)=T(3)
        ED(4*(LENEDT+3))=T(4)
        FMT(IFT)=FMI1
        FMT(IFT+1)=FMI2
        LENEDT=LENEDT+3
90  CONTINUE
      WRITE(NOUT,FMT) (EDIT(JK),JK=1,LENEDT)
100 CONTINUE
      IS=IS+300
      ISTO=ISTO+300
      IF(ISTO.GT.ISTP) ISTO=ISTP
120 CONTINUE
      RETURN
      END

```

Work Done By

H. C. Honeck
W. H. Hightower
C. E. Henderson

Report Written By

H. C. Honeck

Contents

	<u>Page</u>
1. Introduction	B.2.3
2. Conventions and Notation	B.2.6
3. NAME Block	B.2.8
4. CØMP Block	B.2.11
5. GEØT and GEØM Blocks	B.2.13
6. ZØNE Block	B.2.19
7. XSEC Blocks	B.2.21
8. SMAT and PMAT Blocks	B.2.23
9. CHI Block	B.2.27
10. RESD Block	B.2.29
11. TMAT Blocks	B.2.31
12. FLUX, JIN, JØUT, and QCEL Blocks	B.2.35
13. HØMX Block	B.2.37
14. BURN Blocks	B.2.41
15. GRPS Block	B.2.43
16. SRCH and LFLX Blocks	B.2.45
17. CARD Block	B.2.48 thru B.2.58

JOSHUA

DPSTM-500

Title Lattice Intermediate Data Sets (LIDS)

Vol.
IV

Page
B.2.2

Rev.

Date
2/70

(blank)

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.3
		Rev.	Date 2/70

1. Introduction

The lattice physics subsystem interfaces with other subsystems through the permanent data sets ASSEMBLY, MULTIGRP, FINEGRP, RILIB, FEWGRP, and LATTICE. From the user's point of view, the entire subsystem is a single functional module. From a programming point of view, the subsystem is a collection of modules driven by the control module RAHAB. These modules are interfaced through a special data set called TEMP. Since other subsystems will also use the TEMP data set, the name is further qualified by the name LIDS (Lattice Intermediate Data Set). The rationale for permanent versus temporary data sets and their usage is given in the overview of the subsystem and will not be repeated here.

LIDS is composed of individual records called "Blocks". These blocks are read and written by the JOSHUA data manager so that each must have a unique name. All blocks in the LIDS have the name structure:

'TEMP'.'LIDS'.DLAT.VLAT.BLØK.NCASE.N1.N2

DLAT	-	lattice name
VLAT	-	lattice version
BLØK	-	block name
NCASE	-	case number
N1,N2	-	qualifying integers which may be:
		NEB - energy band number
		NZN - spatial zone number
		NBS - buckling search number
		NRG - region number

Blocks are manipulated in-core by the MANAGE subroutine (See Section B.1, Data Management Techniques) and are referred to by the last four qualifiers (BLOK, NCASE, N1,N2).

This manuscript contains the specifications for all of the blocks in the LIDS. The block names are summarized in the following tables

LIDS Blocks Packed into the DATA Array by the MANAGE Routine

<u>Block Name</u> *	<u>Description</u>
C.'NAME'.NCASE.O .O	All 8-character names.
C.'COMP'.NCASE.O .O	Composition data.
C.'GEOT'.NCASE.O .O	Geometry tables (complete).
C.'GEOM'.NCASE.O .O	Geometry tables (partial).
C.'ZONE'.NCASE.O .O	Zone tables.
C.'XSEC'.NCASE.NEB.O	Microscopic cross sections.
C.'SMAT'.NCASE.NEB.O	Microscopic scattering matrices.
C.'PMAT'.NCASE.NEB.O	Macroscopic homogenized scattering matrices.
C.'CHI' .NCASE.O .O	Fission spectra.
C.'TMAT'.NCASE.NEB.NZN	Transport matrices.
C.'FLUX'.NCASE.O .NBS	Flux (sub-region).
C.'FLUX'.NCASE.1 .NBS	Flux (region)
C.'JIN' .NCASE.O .NBS	Inward currents.
C.'JOUT'.NCASE.O .NBS	Outward currents.
C.'QCEL'.NCASE.O .NBS	Cell slowing-down density.
C.'HOMX'.NCASE.O .O	Macroscopic homogenized cross sections.
C.'BURN'.NCASE.NBS.NRG	Reaction rates.
C.'RESO'.NCASE.O .O	Resonance data.
C.'RESO'.NCASE.1 .O	Resonance data corrected for flux fine-structure.

* C = 'TEMP'. 'LIDS'.DLAT.VLAT

LIDS Blocks Not Packed Into the DATA Array

<u>Block Name</u> *	<u>Description</u>
C.'CARD'.NCASE.0 .0	Geometry and composition data
C.'GRPS'.NCASE.0 .0	Energy group structure
C.'SRCH'.NCASE.0 .0	Leakage search results
C.'LFLX'.NCASE.NBS .0	Legendre fluxes

*C = 'TEMP'.'LIDS'.DLAT.VLAT

2. Conventions and Notation

Many conventions have been previously described in manuscripts A.1, A.2, and B.1. In particular, the reader must be familiar with B.1, Data Management Techniques, to understand the block structures given here.

An attempt has been made to use a uniform notation throughout the subsystem. The bulk of the notation follows a very simple set of rules. For example, the following symbols are used:

<u>Symbol</u>	<u>Meaning</u>	<u>Symbol</u>	<u>Meaning</u>
LP	Lattice position	FC	Face
CL	Cell	SD	Sides
SA	Sub-assembly	RD	Radius
RG	Region	VR	Vertex
SR	Sub-region	ZN	Zone
MT	Material	DT	Data Type
IS	Isotope	XS	Cross Section
RI	Resonance Isotope	SM	Scattering Matrix
XT	Total Cross Section	CH	Chi
XF	Fission Cross Section	FI	Fissionable Isotope
NU	Nu	XA	Absorption Cross Section
SF	Resonance Shielding Factor	RA	Resonance Absorption
FG	Few Group	RF	Resonance Fission
BS	Buckling Search		

These symbols are then combined with a set of qualifying symbols.

<u>Symbol</u>	<u>Meaning</u>
I	Prefix denoting a relative count
N	Prefix denoting an absolute count
K	Prefix denoting a pointer
X	Postscript denoting the upper limit of a count
A	Prefix denoting either "area" or "adjacent"
NXT	Next
V	Volume
NAM	Postscript denoting name

Some Examples are:

<u>Symbol</u>	<u>Meaning</u>
ISR	Relative sub-region number (within a region)
NSR	Absolute sub-region number (within the supercell)
NSRX	Number of sub-regions in supercell
KSR	Sub-region pointer
KNXTSR	Pointer to next sub-region in list
AFC	Area of a face
NAFC	Adjacent face number
KISNAM	Pointer to an isotope name

The remaining symbols are defined in the following table:

<u>Symbol</u>	<u>Definition</u>
I	Initial energy group number
J	Final energy group number
IA	Lower I limit of non-zero data
IB	Upper I limit of non-zero data
JA	Lower J limit of non-zero data
JB	Upper J limit of non-zero data
CØNC	Concentration
ITDC	Temperature (°C)
ZAS	ZAS number
AWR	Atomic weight ratio
XR,YR	X and Y coordinates of circle center
XV,YV	X and Y coordinates of vertex
XDISP,YDISP	X and Y displacement for entering adjacent face
RØTANG	Rotation angle for entering adjacent face
NEL	Transport matrix element number
IDBF	Detailed balance flag
NSAK	Sub-assembly kind number
METHØD	Collision probability generator method number

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.8
		Rev.	Date 2/70

3. NAME Block

8-character names are used to designate cells, regions, materials, and isotopes. All character data of this type is packed in the NAME block. To avoid boundary alignment problems the NAME block is always the first block in the DATA array. Since DATA is an array of 4 byte words, two adjacent words are used to store the 8 byte name.

For example, the pointer KMTNAM is stored in the CØMP block. The first 4 characters of the material name are in DATA (KMTNAM) and the last 4 characters are in DATA (KMTNAM+1). Note that no origin is needed since the NAME block is always the first block in the DATA array.

DATA (K)

Block name: 'NAME'.NCASE.O.O

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.10
		Rev.	Date 2/70

(blank)

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.11
		Rev.	Date 2/70

4. CØMP Block

The CØMP block contains the composition data. There are two separate lists packed into the block. The first list describes materials. The principal datum given is the concentration in atoms/barn-cm, CØNC(IIS,NMT), of the IIS isotope in material number NMT. IIS is the count of isotopes in this material, $1 \leq \text{IIS} \leq \text{IISX}$.

The second list contains isotope data; temperature (ITDC), pointer to the isotope name (KISNAM), atomic weight ratio (AWR), and the ZAS number. The name and temperature are used to read data for this isotope from the MULTIGRP data set.

CØMP(K)
KCPT(2,K)Prologue

KCØRG+1	NMTX	KMT1	NØ. ØF MATERIALS, POINTER TØ FIRST MATERIAL
	NISX	KIS1	NØ. ØF ISØTØPES, PØINTER TØ FIRST ISØTØPE

Material Description

KMT+0	NMT	KNXTMT	MATERIAL NØ., PØINTER TØ NEXT MATERIAL
+1	IISX	KMTNAM	NØ. ØF ISØTØPES IN THIS MATERIAL, PØINTER TØ MATERIAL NAME
+2			(not used)
+3	NIS ₁	KIS ₁	ISØTØPE NØ., PØINTER TØ ISØTØPE DESCRIPTION
	CØNC ₁		CØNCENTRATION (ATM/B-CM) ØF THIS ISØTØPE IN THIS MATERIAL
	NIS ₂	KIS ₂	
	CØNC ₂		

	CØNC _{IISX}		

Isotope Description

KIS+0	NIS	KNXTIS	ISØTØPE NØ., PØINTER TØ NEXT ISØTØPE
+1	ITDC	KISNAM	INTEGER T(°C), PØINTER TØ ISØTØPE NAME
+2	AWR		ATØMIC WEIGHT RATIØ
+3	ZAS		ZAS NUMBER
+4	LIXS		IF LIXS≠0, IGNØRE XSECTION S IN SPECTRUM CALCULATION

Block name: 'CØMP'.NCASE.0.0

5. GEØT and GEØM Blocks

The GEØT block contains the data for the two-dimensional super-cell geometry as described in manuscript IV.C.2. The GEØT block contains 8 separate lists (describing cells, sub-assemblies, regions, sub-regions, faces, sides, radii, and vertices) plus a prologue which tells where each list begins.

The GEØM block is an abbreviated version of the GEØT block and contains only the first 4 lists (describing cells, sub-assemblies, regions, and sub-regions) plus the prologue.

The full block names are:

'GEØT'.NCASE.O.O

'GEØM'.NCASE.O.O

GEOT(K)
KGPT(2,K)

Prologue

KGØRG+1	NCLX	KCL1
2	NSAX	KSA1
3	NRGX	KRG1
4	NSRX	KSR1
5	NFCX	KFC1
6	NSDX	KSD1
7	NRDX	KRD1
8	NVRX	KVR1
9	NLPX	

NØ. ØF CELLS, PØINTER TØ FIRST CELL
 NØ. ØF SUB-ASSEMBLIES, PØINTER TØ FIRST SUB-ASSEMBLY
 NØ. ØF REGIØNS, PØINTER TØ FIRST REGIØN
 NØ. ØF SUB-REGIØNS, PØINTER TØ FIRST SUB-REGIØN
 NØ. ØF FACES, PØINTER TØ FIRST FACE
 NØ. ØF SIDES, PØINTER TØ FIRST SIDE
 NØ. ØF RADII, PØINTER TØ FIRST RADIUS
 NØ. ØF VERTICES, PØINTER TØ FIRST VERTEX
 NØ. ØF LATTICE PØSITIØNS IN SUPER CELL

Cell Description

KCL+0	NCL	KNXTCL
1	ISAX	KCLNAM
2	NSA ₁	KSA ₁
	NSA ₂	KSA ₂

	NSA _{ISAX}	KSA _{ISAX}
	ILPX	
	NLP ₁	
	NLP ₂	

	NLP _{ILPX}	

CELL NØ., PØINTER TØ NEXT CELL
 NØ. ØF SUB-ASSEMBLIES IN CELL, PØINTER TO CELL NAME

SUB-ASSEMBLY NØS. AND PØINTERS

NØ. OF LATTICE PØSITIØNS FØR THIS CELL

LATTICE PØSITIØN NUMBERS

GEOT(K)
KGPT(2,K)Sub-Assembly Description

KSA+0	NSA	KNXTSA	SUB-ASSEMBLY NO., POINTER TO NEXT SUB-ASSEMBLY
1	NCL	KCL	ASSOCIATED CELL NO. AND POINTER
2	NSAK	NZN	SUB-ASSEMBLY KIND NUMBER, ZONE NO.
3	IRGX	NAZM	NO. OF REGIONS IN SUB-ASSEMBLY, NO. OF AZIMUTHAL SUBDIVISIONS
4	NRG ₁	KRG ₁	} REGION NOS. AND POINTERS
	NRG ₂	KRG ₂	

	NRG _{IRGX}	KRG _{IRGX}	

Region Description

KRG+0	NRG	KNXTRG	REGION NO., POINTER TO NEXT REGION
1	NSA	KSA	ASSOCIATED SUB-ASSEMBLY NO. AND POINTER
2	VRG		REGION VOLUME
3	ISRX	KRGNAM	NO. OF SUB-REGIONS IN REGION, POINTER TO REGION NAME
4	NSR ₁	KSR ₁	} SUB-REGION NOS. AND POINTERS
	NSR ₂	KSR ₂	

	NSR _{ISRX}	KSR _{ISRX}	

Sub-Region Description

KSR+0	NSR	KNXTSR	SUB-REGION NO., POINTER TO NEXT SUB-REGION
1	NRG.	KRG	ASSOCIATED REGION NO. AND POINTER
2	VSR		SUB-REGION VOLUME
3	ISDX	KMT	NO. OF SIDES ON SUB-REGION, POINTER TO MATERIAL (IN COMP)
4	NSD ₁	KSD ₁	} SIDE NOS. AND POINTERS. (Note: These are omitted in the GEOM block.)
	NSD ₂	KSD ₂	

	NSD _{ISDX}	KSD _{ISDX}	

GEOT(K) KGPT(2,K)		Face Description	
KFC+0	NFC	KNXTFC	FACE NØ., PØINTER TO NEXT FACE
1	NSR	KSR	ASSØCIATED SUB-REGION NØ. AND PØINTER
2	NSD	KSD	ASSØCIATED SIDE NØ. AND PØINTER
3	AFC		FACE AREA
4	ILPX		NØ. ØF LATTICE PØITIONS FØR THIS CELL
KLP+0	NLP	KNXTLP	LATTICE PØITION*, PØINTER TØ NEXT PØITION
1	XDISP		X DISPLACEMENT FØR ENTERING ADJACENT FACE
2	YDISP		Y DISPLACEMENT FØR ENTERING ADJACENT FACE
3	RØTANG		RØTATION ANGLE FØR ENTERING ADJACENT FACE
4	IAFCX	NALP	} ADJACENT FACE NØS. AND PØINTERS
5	NAFC ₁	KAFC ₁	
	NAFC ₂	KAFC ₂	

	NAFC IAFCX	KAFC IAFCX	

* NLP=0 denotes an internal face of a cell.

<u>Side Description</u>		
KSD+0	NSD	KNXTSD
1	NSR ₁	KSR ₁
2	NSR ₂	KSR ₂
3	NFC	KFC
4	NRD	KRD
5	NVR ₁	KVR ₁
6	NVR ₂	KVR ₂

SIDE NØ., PØINTER TO NEXT SIDE

SUB-REGION NØS. AND PØINTERS SEPARATED
BY THIS SIDE**

ASSØCIATED FACE NØ. AND PØINTER (Zero if not a face.)

RADIUS NØ. AND PØINTER (Zero if a linear side.)

VERTEX NØS. AND PØINTERS DEFINING
THIS SIDE (Zero if a full circle.)

**If this side is a circular arc, NSR₁ refers to the sub-region inside the circle, and NSR₂ refers to the sub-region outside the circle. If this side is a face, one of the two sub-region numbers will be zero.

GEOT(K)
KGPT(2,K)

Radius Description

KRD+0	NRD	KNXTRD
1	RD	
2	XR	
3	YR	

RADIUS NO., POINTER TO NEXT RADIUS

RADIUS

} X AND Y OF CIRCLE CENTER*

Vertex Description

KVR+0	NVR	KNXTVR
1	XV	
2	YV	

VERTEX NO., POINTER TO NEXT VERTEX

X COORDINATE*

Y COORDINATE*

* Coordinates are relative to the center of the cell.

JOSHUA
DPSTM-500

Title Lattice Intermediate Data Sets (LIDS)

Vol.
IV

Page
B.2.18

Rev.

Date
2/70

(Blank)

6. ZØNE Block

The ZØNE block is a collection of selected geometry data needed by the module which solves the matrix transport equation. The data is organized by zone. For each zone (NZN) there are two lists. The first is a list of sub-region numbers (NSR), volumes (VSR), and material pointers (KMT) for the sub-regions in the zone. The second is a list of faces (NFC) of the zone. For each face, the face area (AFC) and a list of adjacent faces (NAFC) and current fractions (F(NFC←NAFC)) are given. These fractions are the geometry boundary conditions. If JIN and JØUT represent inward and outward neutron currents, they are related by:

$$JIN(NFC) = \sum_{NAFC} F(NFC \leftarrow NAFC) * JØUT(NAFC)$$

ZONE(K)
KZPT(2,K)

KZØRG+1	NZNX	
KZN+0	NZN	KNXTZN
1	METHØD	KIFCX
2	ISRX	NCL
3	NSR ₁	KMT ₁
	VSR ₁	

	NSR _{ISRX}	KMT _{ISRX}
	VSR _{ISRX}	
KIFCX	IFCX	
KFC+0	NFC ₁	KNXTFC
1	AFC	
2	IAFCX	
3	NAFC ₁	
	F(1 ← 1)	
	NAFC ₂	
	F(1 ← 2)	

	NAFC _{IFCX}	
	F(1 ← IAFCX)	

NØ. ØF ZØNES

ZØNE NØ., PØINTER TØ NEXT ZØNE

CØLLISIØN PRØBABILITY METHØD NØ., PØINTER TØ IFCX

NØ. ØF SUB-REGIØNS, CELL NØ.

SUB-REGIØN NØ., PØINTER TØ MATERIAL (IN CØMP)
SUB-REGIØN VØLUME

NØ. ØF FACES IN ZØNE

FIRST FACE NØ., PØINTER TØ NEXT FACE

FACE AREA

NØ. ØF FACES ADJACENT TØ FIRST FACE

ADJACENT FACE NØ.

FRACTIØN OF CURRENT LEAVING ADJACENT
FACE WHICH ENTERS THIS FACEBlock name: 'ZONE'.NCASE.0.0

7. XSEC Blocks

The XSEC blocks contain microscopic cross section data by isotope, data (reaction) type, and energy group. The data is given in a series of XSEC blocks, one for each energy band. Only non-zero data is given. These XSEC blocks are packed forms of the data on the MULTIGRP data set.

XSEC(K)
KXPT(2,K)Prologue for Each Isotope

KXS+0		KNXTXS
1	NIS	KIS
2		KDT1
3		KDT2
4		KDT3
5		KDT4
6		KDT5
7		KDT6
8		KDT7
9		KDT8
10		KDT9
11		KDT10
12		KDT11
13		KDT12
14		KDT13
15		KDT14
16		KDT15
17		KDT16

PØINTER TØ CRØSS SECTIONS FØR NEXT ISØTØPE
ISØTØPE NØ. AND PØINTER TØ CØMP BLØCK

PØINTER TØ DATA TYPE

1	(TØTAL)
2	(CAPTURE)
3	(FISSION)
4	(ν)
5	(TRANSPORT)
6	($n, 2n$)
7	(not used)
8	(n, ν)
9	(n, α)
10	(n, p)
11	(n, D)
12	(n, T)
13	(n, ν^*)
14	
15	
16	

Cross Section Data

KDT+0	IA	IB
1	$\sigma(\text{IA})$	
2	$\sigma(\text{IA}+1)$	

	$\sigma(\text{IB})$	

LØWER AND UPPER GRØUP LIMITS

CRØSS SECTIØN (BARNS) IN GRØUP IACRØSS SECTIØN (BARNS) IN GRØUP IA+1

CRØSS SECTIØN (BARNS) IN GRØUP IBBlock name: 'XSEC'.NCASE.NEB.0

8. SMAT and PMAT Blocks

The SMAT blocks contain microscopic scattering matrix data by isotope, data (reaction) type, and group. The data is given in a series of SMAT blocks, one for each band of final energies. Only non-zero data is given. The SMAT blocks are packed forms of the data on the MULTIGRP data set.

The data included is the scattering matrix, $\sigma_L(I \rightarrow J)$, where L is the Legendre order ($0 \leq L \leq LX$), I is the initial group number, and J is the final group number (see Figure 1). Also given is the associated cross section, $\sigma_L(J)$, defined by

$$\sigma_L(J) = \sum \sigma_L(J \rightarrow I) * DE(I)$$

where $DE(I)$ is the group width in ev. Note that the matrix elements have units of barns/ev. The Legendre components are defined by

$$\frac{d\sigma(I \rightarrow J)}{d\mu} = \sum_{L=0}^{LX} \frac{2L+1}{2\pi} \sigma_L(I \rightarrow J) P_L(\mu)$$

The PMAT block contains macroscopic scattering matrix data for a homogeneous media and is identical in structure to the SMAT blocks with the following exceptions:

1. IDBF, KNXTSM, NIS, and KIS are all zero since there is only one "isotope" (the homogeneous media), and upscattering is always given explicitly.
2. The data are macroscopic cross section (cm^{-1}) and matrix elements ($\text{cm}^{-1}\text{ev}^{-1}$)

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.24
		Rev.	Date 2/70

Figure 1. Schematic of a Scattering Matrix, $\sigma_L(I \rightarrow J)$

SMAT(K)
KSPT(2,K)

Prologue for Each Isotope

KSM+0	IDBF	KNXTSM	IDBF=0 (FULL MATRIX),=1 (HALF), POINTER TO DATA FOR NEXT ISOTOPE
1	NIS	KIS	ISOTOPE NO. AND POINTER TO COMP TABLE
2		KDT1	POINTER TO DATA TYPE 1 (ELASTIC)
3		KDT2	2 (INELASTIC)
4		KDT3	3 (n,2n)
5		KDT4	4 (TOTAL=EL+IN+2*N2N)
6		KDT5	5
7		KDT6	6
8		KDT7	7
9		KDT8	8

Scattering Matrix Data

KDT	JA	JB	LOWER AND UPPER FINAL GROUP LIMITS
KJ=KDT+1	J	KNXTJ	FINAL GROUP NO., POINTER TO NEXT J
KJ+1	LX	IX	MAX. LEGENDRE ORDER GIVEN, $IX=IB-IA+2$
+2	IA	IB	LOWER AND UPPER INITIAL GROUP
+3	$\sigma_0(J)$		SCATTERING MATRIX DATA AND CROSS SECTION
	$\sigma_0(IA \rightarrow J)$		IF $IA \leq I \leq IB$ AND $L \leq LX$
	$\sigma_0(IA+1 \rightarrow J)$		THE CROSS SECTION IS AT:
	----		$K = KJ + L*IX+3$
	$\sigma_0(IB \rightarrow J)$		MATRIX ELEMENTS ARE AT:
	$\sigma_1(J)$		$K = KJ + L*IX + I - IA + 4$
	$\sigma_1(IA \rightarrow J)$		
	$\sigma_1(IA + 1 \rightarrow J)$		

	$\sigma_1(IB \rightarrow J)$		

	$\sigma_{LX}(IB \rightarrow J)$		

Block name: 'SMAT'.NCASE.NEB.O

JOSHUA
DPSTM-500

Title Lattice Intermediate Data Sets (LIDS)

Vol. IV Page B.2.26

Rev. Date 2/70

(Blank)

9. CHI Block

The CHI block contains the fission spectrum (independent of incident energy) by isotope and group. The data is normalized such that

$$\sum_I \text{CHI}(I, \text{NIS}) * \text{DE}(I) = 1$$

KCH+O

CHIT(K) KCPT(2,K)	
NIS	KNXTCH
IA	IB
CHI(IA)	
CHI(IA+1)	

CHI(IB)	

ISOTOPE NUMBER, POINTER TO NEXT ISOTOPE
 LOWER AND UPPER GROUP LIMITS
 FISSION SPECTRUM FOR GROUP $IA \leq I \leq IB$

Block name: 'CHI'.NCASE.0.0

10. RESD Block

The RESD block contains resonance absorption rates, fission rates, and self-shielding factors by isotope, region, and energy group. The rates have units of total reaction in the region per unit total slowing density (neutrons/sec) in the cell containing the region. The self-shielding factor is the ratio of the actual resonance absorption rate to the infinite dilute absorption rate.

RESD(K)
KPRT(2,K)

KRØRG+1	IA	IB	LOWER AND UPPER GRØUP LIMITS
KI+0	I	KNXTI	GRØUP NUMBER, PØINTER TØ NEXT GRØUP
+1	NRIX		NUMBER ØF ISØTØPE/REGION PAIRS
KRI+0	NIS	NRG	ISØTØPE NUMBER, REGION NUMBER
+1	RA(NIS,NRG,I)		RESØNANCE ABSØRPTION, FISSION, AND SHIELDING FACTØR FØR THIS ISØTØPE/REGION AND GRØUP
+2	RF(NIS,NRG,I)		
+3	SR(NIS,NRG,I)		

Block name: 'RESD'.NCASE.0.0

11. TMAT Blocks

The TMAT blocks contain the transport matrix data by energy group. A separate block is used for each energy band (NEB) and zone (NZN). A schematic of a transport matrix is shown in Figure 2. The TMAT block is in two parts. The first part is a table which relates the element pointers NEL with the indices ISF, JSF. The second part of the block contains the distinct matrix elements by group, T(NEL,I).

$$\begin{aligned}
 ISF &= ISR, & ISF &\leq ISRX \\
 &= ISRX + IFC, & ISF &> ISRX \\
 ISFX &= ISRX + IFCX
 \end{aligned}$$

Figure 2. Schematic of a Transport Matrix

TMAT(K)
KTPT(2,K)

KTM+0	NZN	
1	ISFX	KNELX
2	NEL(1)	NEL(2)
	NEL(3)	NEL(4)

KNEL	NEL(L)	NEL(L+1)

	NEL(LX-1)	NEL(LX)
KNELX+0	NELX	
+1	IA	IB
KI	I	KNXTI
	T(1,I)	
	T(2,I)	

KT	T(NEL,I)	

	T(NELX,I)	

ZONE NO.

NO. OF SUB-REGS + FACES, POINTER TO NELX

POINTERS TO THE DATA BELOW

$$LX = (ISFX)^2$$

(SEE BELOW)

NO. OF DISTINCT ELEMENTS

LOWER AND UPPER GROUP LIMITS

GROUP NO., POINTER TO NEXT GROUP

THE DISTINCT MATRIX ELEMENTS FOR THIS GROUP

$$L = ISF + ISFX*(JSF-1)$$

$$KRL = L - 2*((L-1)/2)$$

$$KNEL = KTM + 2 + (L-1)/2$$

$$NEL = KTPT(KRL,KNEL)$$

$$KT = KI + NEL$$

Record name: 'TMAT'.NCASE.NEB.NZN

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.34
		Rev.	Date 2/70

(Blank)

12. FLUX, JIN, JØUT and QCEL Blocks

Each of these blocks is a simple two dimensional dense array of numbers so that the structure can be easily described by a simple Fortran list. For example, the block named 'FLUX'. NCASE.O.NBS corresponds to the list

$$((F(NSR,I),NSR = 1,NSRX) , I = 1,NEG)$$

where NSR is the sub-region index and I is the group index. The block is read by the MANAGE routine into the DATA array and assigned an origin (say KFØRG). The flux for a given NSR and I is located at $K = NSR + NSRX*(I-1)$. In this example, NSR is called the inner index, and I the outer index. Since each block has a similar structure, the attributes can be summarized in the accompanying table.

Fluxes have units of neutrons/cm²sec ev and may be stored both by sub-region ('FLUX'.NCASE.O.NBS) and by region ('FLUX'.NCASE.1.NBS). The blocks JIN and JØUT contain inward and outward neutron currents (neutrons/cm²sec ev) for each face (NFC) and group (I). The blocks QCEL contain the total neutrons slowing down in a cell (neutrons/sec) past energy E(I) for each cell (NCL).

The symbol NBS in the name designates the Buckling search number, since each block may be based on a different leakage weighting spectrum.

Block name	Array Symbol	Origin Symbol	Inner Variable	Outer Variable	One dimensional array index
'FLUX'.NCASE.O.NBS	FLUX	KFØRG	NSR	I	$K=NSR+NSRX*(I-1)$
'FLUX'.NCASE.1.NBS	FLUX	KFØRG	NRG	I	$K=NRG+NRGX*(I-1)$
'JIN'.NCASE.O.NBS	JIN	KIØRG	NFC	I	$K=NFC+NFCX*(I-1)$
'JØUT'.NCASE.O.NBS	JØUT	KØØRG	NFC	I	$K=NFC+NFCX*(I-1)$
'QCEL'.NCASE.O.NBS	QCEL	KQØRG	NCL	I	$K=NCL+NCLX*(I-1)$

13. HØMX Block

The HØMX block contains macroscopic cross sections and resonance reaction rates for a homogeneous medium. The block contains two parts. The first part lists the total cross section (cm^{-1}), absorption cross section (cm^{-1}), and resonance absorption rate (absorptions per unit slowing down density) for each energy group. The second part lists the fission cross section (cm^{-1}), resonance fission rate (fissions per unit slowing down density), neutrons per fission, and normalized fission spectrum (ev^{-1}) for each group and for each fissionable isotope.

HØMX(K)
KHPT(2,K)

KHØRG+1	NFIX	KFI1
+2		KXT
+3		KXA
+4		KRA
KXT	XTØT(1)	
	XTØT(2)	

KXA	XTØT(NEG)	
	XABS(1)	

KRA	XABS(NEG)	
	RABS(1)	

	RABS(NEG)	

NO. OF FISSIONABLE ISOTOPES, POINTER TO 1ST ISOTOPE

POINTER TO TOTAL CROSS SECTION

POINTER TO ABSORPTION CROSS SECTION

POINTER TO RESONANCE ABSORPTION RATES

TOTAL CROSS SECTION FOR MIXTURE BY GROUP

ABSORPTION CROSS SECTION FOR MIXTURE BY GROUP

RESONANCE ABSORPTION RATE FOR MIXTURE BY GROUP

Block name: 'HØMX'.NCASE.O.O

HØMX(K) KHPT(2,K)		
KFI	NFI	KNXTFI
+1	NIS	KXF
+2		KNU
+3		KRF
+4		KCH
KXF	XFIS(1)	} FISSION CROSS SECTION FOR ISOTOPE BY GROUP

	XFIS(NEG)	
KRF	RFIS(1)	} RESONANCE FISSION RATES FOR ISOTOPE BY GROUP

	RFIS (NEG)	
KNU	XNU(1)	} NU FOR ISOTOPE BY GROUP

	XNU(NEG)	
KCH	CHI(1)	} CHI FOR ISOTOPE BY GROUP

	CHI(NEG)	

FISS. ISOTOPE NO., POINTER TO NEXT FISSION ISOTOPE
 ISOTOPE NO. (IN 'COMP'), POINTER TO FISSION CROSS SECTION
 POINTER TO NU
 POINTER TO RESONANCE FISSION RATES
 POINTER TO CHI

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.40
		Rev.	Date 2/70

(Blank)

14. BURN Blocks

The BURN blocks contain few group fluxes and microscopic few group cross sections for each isotope and reaction type in a region. A separate BURN block is used for each region (NRG) in a supercell. In addition, separate BURN blocks are used for each Buckling search number (NBS), that is, for each leakage flux weighting spectrum used. The data is given such that the total number of reactions of type NDT in isotope IIS in a region is given by

$$\text{Reactions/sec} = \sum_{\text{NFG}=1}^{\text{NFGX}} \text{CONC(IIS)} * \sigma(\text{NFG}, \text{NDT}, \text{IIS}) * \text{FLX}(\text{NFG})$$

The data type numbers are defined in the following table.

<u>NDT</u>	<u>Reaction Type</u>
1	(n,γ) - Smooth
2	(n,γ) - Resonance
3	Fission - Smooth
4	Fission - Resonance
5	Neutrons/fission
6	Resonance absorption shielding factor
7	(n,α)
8	(n,p)
9	(n,γ) to first excited state
10	(n,2n)

BURN(K) KBPT(2,K)		
KBØRG+1	NMT	KRGNAM
2	NFGX	KFSTIS
3	VRG	
4	FLX(1)	FEW GROUP TOTAL FLUXES (NEUTRON -CM/SEC)
	FLX(2)	

	FLX(NFGX)	
KIS+0	IIS	KNXTIS
1	CØNC(IIS)	
KDT+0	NDT	KNXTDT
1	σ(1,NDT,IIS)	MICROSCOPIC CROSS SECTION (BARNS) BY FEW GROUP FOR THIS DATA TYPE AND ISOTOPE.
	σ(2,NDT,IIS)	
	σ(NFGX,NDT,IIS)	

Block name: 'BURN'.NCASE.NBS.NRG

15. GRPS Block

The GRPS block contains group structure data. Because of its small size and frequency of use, it is assigned its own ~~CØMMØN~~ area and not included in the DATA array.

~~CØMMØN~~/GRØUPS/NEG,JFTG,JCHI,IDBF,E(101),EBAR(100),
DE(100),NEBX,IEBH(100)

- NEG - Number of energy groups
- JFTG - First thermal group number
- JCHI - Last group where the fission spectrum is non-zero.
- IDBF - Detailed Balance flag. If IDBF = 0, full scattering matrix data is given on the MULTIGRP data set. If IDBF = 1, only downscattering is given and upscattering is computed from Detailed Balance.
- E(I) - Upper energy limit (ev) of group I
- EBAR(I) - Characteristic energy (ev) of group I
- DE(I) - Energy width (ev) of group I
- I - Group index. I = 1 refers to the highest energy group and I = NEG to the lowest energy group
- J - Energy band index, J = 1,NEBX
- NEBX - Number of energy bands
- IEBH(J) - Upper group index I included in energy band J.

The GRPS block is read into this ~~CØMMØN~~ area with a JOSHUA read statement using:

Name: 'LIDS'.DLAT.VLAT.'GRPS'.NCASE.0.0

List: NEG,NEBX,JFTG,JCHI,IDBF,(E(I),EBAR(I),DE(I),I=1,NEG),E(NEG+1),(IEBH(J),J=1,NEBX)

where DLAT and VLAT are the lattice name and version.

Most of this information comes directly from the MULTIGRP data set. The band structure is determined at execution time and will depend on the size of the problem being run.

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.44
		Rev.	Date 2/70

(Blank)

16. SRCH and LFLX Blocks

Following homogenization of a lattice, a variety of B_l leakage calculations may be run. The resulting Legendre flux is used to correct the cell fine-structure flux for leakage effects. The SRCH and LFLX Blocks contain the results of these B_l calculations.

Name: 'LIDS'.DLAT.VLAT.'SRCH'.NCASE.O.O

List: NBSX,(LSØP(NBS),KEFF(NBS),BSQD(NBS),
MSQD(NBS),ALPHA(NBS),NBS=1,NBSX)

Name: 'LIDS'.DVAT.VLAT.'LFLX'.NCASE.NBS.O

List: NEG,NPX(QSD(I),DLKG(I),(PHI(NP,I),
NP=1,NPX),I=1,NEG

DLAT	-	Lattice name
VLAT	-	Lattice version
NCASE	-	Case number
NBS	-	Buckling search number
NBSX	-	Number of searches done
NP	-	Legendre index (=1+Legendre order)
NPX	-	Maximum Legendre index
I	-	Energy group number
NEG	-	Number of energy groups
LSØP	-	Search option
		1 - k_{eff} calculation
		2 - buckling search
		3 - time eigenvalue search
KEFF	-	k_{eff}
BSQD	-	Buckling (cm^{-2})
MSQD	-	Migration Area (cm^2)

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol.	Page
		IV	B.2.46
		Rev.	Date
			2/70

ALPHA - Time eigenvalue (sec^{-1})
PHI(I,NP) - Legendre flux (neutrons/ cm^2sec) in
 group I of order NP-1
DLKG(I) - Diffusion coefficient (cm) in group I
QSD(I) - Slowing down density (neutron/sec)

17. CARD Block

The CARD block contains geometry and composition data in a structure similar to a deck of cards. In the early stages of the development of the lattice physics subsystem, several modules were prepared using card input. Most of these were later converted to binary records as the module interfaces became better understood. The CARD block was also made into a binary record but its internal card structure was retained. This allows one to prepare unusual geometry descriptions in card form and run Monte Carlo calculations.

The CARD block is read with JOSHUA read statements using:

Name: 'TEMP'.'LIDS'.DLAT.VLAT.'CARD'.NCASE.O.O

List: NX,(CARD(I),I=1,NX)

where DLAT.VLAT is the lattice name and version, and NCASE is the case number. The array CARD is equivalenced to an integer array KARD so that a field may be either REAL or INTEGER depending on its context. In describing the structure of the CARD block, we will visualize it as a deck of cards with 6 fields per card. The basic structure is illustrated on the next page and detailed fields descriptions are given on the following pages. The reader should refer to manuscript A.2 for a description of the geometry and materials.

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.48
		Rev.	Date 2/70

Supercell Header Card

Composition Pointer Card

Geometry Header Card

Sub-assembly Header Card, sub-assembly 1

(Geometry Deck for sub-assembly 1)

Sub-assembly Header Card, sub-assembly 2

(Geometry Deck for sub-assembly 2)

Cell 1

- - - - -

- - - - -

Geometry Pointer Card

Composition Header Card

(Material Deck for material 1)

(Material Deck for material 2)

- - - - -

Composition Pointer Card

Geometry Header Card

Cell 2

- - - - -

Header and Pointer Cards

Card	Field	Symbol	Description	* I or R
Supercell Header	1	NCLX	Number of cells in supercell	I
	2	NLPX	Number of lattice positions in supercell	I
Geometry Header	1	NCL	Cell number	I
	2	KCLNAM	Pointer to cell name in NAME block	I
	3	ISAX	Number of sub-assemblies in cell	I
	4	ILPX	Number of lattice positions for this cell	I
Sub-assembly Header	1	ISA	Sub-assembly number within cell	I
	2	NSAK	Sub-assembly kind number	I
	3	NZN	Zone number	I
	4	NAZM	Number of azimuthal subdivisions	I
	4	METH	Collision probability method number	I
Composition Header	1	NCL	Cell number	I
	2	KCLNAM	Pointer to cell name in NAME block	I
	3	IMTX	Number of materials in this cell	I
Geometry pointer	1	KGPT	-KGPT is the location in the CARD array of the Geometry Header Card for the next cell	I
Composition pointer	1	KCPT	-KCPT is the location in the CARD array of the Composition Header Card for the next cell.	I

*I or R denotes Integer or Real

Material Deck

Card	Field	Symbol	Description	I or R
Material	1	IMT	Material number within cell	I
	2	KMTNAM	Pointer to material name in NAME block	I
	3	IISX	Number of isotopes in this material	I
Isotope (Repeat for all isotopes in material)	1	IIS	Isotope number within material	I
	2	KISNAM	Pointer to isotope name in NAME block	I
	3	ITDC	Integer Temperature (°C)	I
	4	ZAS	Z,A,State number	R
	5	AWR	Atomic weight ratio	R
	6	CONC	Concentration (atm/b-cm)	R

Sub-assembly Kind 1
Arbitrary Geometry

Card	Field	Symbol	Description	I or R
Geometry Sub-assembly Header (Kind=1)	1	IRG	Number of regions in this sub-assembly	I
	2	ISRX	Number of sub-regions in this sub-assembly	I
	3	IFCX	Number of faces in this sub-assembly	I
	4	ISDX	Number of sides in this sub-assembly	I
	5	IRDX	Number of radii in this sub-assembly	I
	6	IVRX	Number of vertices in this sub-assembly	I
Region Data (Repeat for all regions)	1	IRG	Region number within sub-assembly	I
	2	KRGNAM	Pointer to region name within NAME block	I
	3	KMTNAM	Pointer to material name within NAME block	I
	4	NSRRG	No. of sub-regions in this region	I
Sub-region Data (Repeat for all sub-regions)	1	ISR	Sub-region number within sub-assembly	I
	2	IRG	Region number in which this sub-region belongs. (this sub-assembly)	I

Sub-assembly Kind 1 (Continued)
Arbitrary Geometry

Card	Field	Symbol	Description	I or R
Sub-region Data (Repeat for all sub-regions)	3	VSR	Volume of this sub-region	R
	4	NSDSR	Number of sides bounding this sub-region	I
Face Data (Repeat for all faces)	1	IFC	Face number in this sub-assembly	I
	2	ISD	Side number of this face in this sub-assembly	I
	3	AFC	Area of this face	R
	4	ILPX	No. of lattice positions that this face will appear	I
	5	IOFT	Inner/outer face (0/1 respectively)	I
Lattice Position Data (Repeat for each face ILPX times)	1	ILP	Lattice position number of face IFC	I
	2	NLP	Lattice position of this cell	I
	3	NALP	Adjacent lattice position of this cell	I
	4	XDISP	X-Displacement	R
	5	YDISP	Y-Displacement	R
	6	ROTANG	Rotation angle (degrees)	R

Sub-assembly Kind 1 (Continued)
Arbitrary Geometry

Card	Field	Symbol	Description	I or R
Side Data (Repeat for each side)	1	ISD	Side Number in this sub-assembly	I
	2	ISRI	The innermost sub-region of this side	I
	3	ISR2	The outermost sub-region of this side	I
	4	IRD	The radius number associated with this side of this sub-assembly	I
	5	IVRI	The first vertex number of this side of this sub-assembly.*	I
	6	IVR2	The second vertex number of this side of this sub-assembly.*	I
Radius Data (Repeat for all radii)	1	IRD	Radius Number within sub-assembly	I
	2	RD	Radius (in cm)	R
	3	XR	X-coordinate of center of imaginary circle	R
	4	YR	Y-coordinate of center of imaginary circle	R
Vertice Data (Repeat for all vertices)	1	IVR	Vertex number within sub-assembly	I
	2	XV	X-coordinate of vertice in this cell	R
	3	YV	Y-coordinate of vertice in this cell	R

* If these vertices define a radial arc (IRD $\neq$ 0), list vertices in counterclockwise direction.

Sub-assembly Kind 2 (or 3)
Annular Geometry (Azimuthally Sub-divided)

Card	Field	Symbol	Description	I or R
Geometry Sub-assembly Header (Kind = 2 or 3)	1	IRGX	Number of regions in this sub-assembly	I
	2	RO	Inner Radius of this sub-assembly	R
Region Data (Repeat for all regions)	1	IRG	Region number within sub-assembly	I
	2	KRGNAM	Pointer to region name within NAME block	I
	3	KMTNAM	Pointer to Material name within NAME block	I
	4	RADØT	Outer radius of this region	R
	5	ISRX	Number of radial sub-divisions of this region	I

Sub-assembly Kind 4 (5, 8 or 9)
Hexagonal or Square Geometry

Card	Field	Symbol	Description	I or R
Geometry Sub-assembly Header (Kind = 4 5, 8 or 9.)	1	IRGX	Number of regions in this sub-assembly	I
	2	RO	Inner Radius of this sub-assembly	R
Region Data (Repeat for all regions.)	1	IRG	Region number within sub-assembly	I
	2	KRGNAM	Pointer to region name within NAME block	I
	3	KMTNAM	Pointer to material name within NAME block	I
	4	HPT	Half pitch of this region	R
	5	ISRX	Number of radial sub-divisions of this region	I
Lattice Number (Repeat for each lattice position.)	1	NLP	Lattice position number for each cell of this super cell that this sub-assembly appears	I
Adjacent Lattice Position Number (Repeat for each face and for each lattice number.)	1	NALP	Adjacent lattice position number of the adjacent lattice position of this sub-assembly	I
	2	IAFC	Adjacent face number of the adjacent lattice position face	I

Sub-assembly Kind 6
Cluster Geometry

Card	Field	Symbol	Description	I or R
Geometry Sub-assembly Header (Kind = 6)	1	IRØDX	Number of rod types in this sub-assembly	I
	2	ICENX	Number of unique rod centers	I
	3	KMDNAM	Pointer to moderator region name within NAME block	I
	4	KMDMAT	Pointer to moderator material name within NAME block	I
	5	CØR	Moderator region outer radius	R
	6	IØPTN	Radial and azimuthal sub-division option	I
Rod Core Data (Repeat for each rod type)	1	IRØDP	Rod core number in this sub-assembly	I
	2	KCRNAM	Pointer to core region name within NAME block	I
	3	KCRMAT	Pointer to core material name within NAME block	I
	4	CRØR	Core region outer radius	R
	5	NRØDX	Number of rods on this radius (evenly spaced)	I
	6	SANGL	Position of first rod	R

Sub-assembly Kind 6(Continued)
Cluster Geometry

Card	Field	Symbol	Description	I or R
Rod Cladding Data (Repeat for each rod type)	1	IRØDP	Rod cladding number in this sub-assembly	I
	2	KCLNAM	Pointer to cladding region name within NAME block	I
	3	KCLMAT	Pointer to cladding material name within NAME block	I
	4	CLØR	Cladding outer radius	R
	5	(blank)		
	6	RSRAD	Radius of rod centers	R

JOSHUA DPSTM-500	Title Lattice Intermediate Data Sets (LIDS)	Vol. IV	Page B.2.58
		Rev.	Date 2/70

(Blank)

JOSHUA
DPSTM-500

Title Part C - Geometry Modules

Vol.
IV

Page
C.O.

Rev.

Date
6/70

Contents

- C.1 Assembly Data Processor - Module ASP
- C.2 Lattice Geometry Routine - Module LAGER
- C.3 Card-loading Assembly Processor - Module CLASP

JOSHUA
DPSTM-500

Title Part C - Geometry Modules

Vol.
IV

Page
C.O.2

Rev.

Date
6/70

(Blank)

JOSHUA

DPSTM-500

Title Assembly Data Processor - Module ASP

Vol.

IV

Page

C.1.1

Rev.

Date

6/70

Work Done By

W. H. Hightower
H. C. Honeck
C. E. Parker

Report Written By

W. H. Hightower
H. C. Honeck

Contents

	<u>Page</u>
1. Theory	C.1.3
2. Program Logic (not included)	
3. Coding Details (not included)	
4. User Information	C.1.4
4.1 Input	C.1.4
4.2 Output	C.1.5
4.3 Core Size and Timing	C.1.5
4.4 Error Stops	C.1.5

JOSHUA

DPSTM-500

Title Module ASP

Vol. IV

Page C.1.2

Rev.

Date 6/70

(Blank)

1. Theory

The ASP module assists the user in setting up the description of a lattice. Individual cells of the lattice are described on the ASSEMBLY Data Set in engineering terms (outer diameters in inches, concentrations in a variety of units). The input record to ASP describes what assemblies are to be used in the lattice, how they are to be put together (the lattice pattern), and how they are to be modified (subdivide regions into sub-regions, homogenize several regions into one region, etc.) for this run. ASP then performs the following operations:

1. Reads data from its input record, and the ASSEMBLY, MULTIGRP, NUCPARAM, and LATTICE Data Sets.
2. Changes dimensions and concentrations to standard units of cm and atoms/barn-cm.
3. Modifies regions by sub-division or homogenization.
4. Reads boundary condition data for the requested lattice pattern from the LATTICE Data Set.
5. Combines all of the data for the lattice into the CARD and NAME blocks and writes them on the Lattice Intermediate Data Set (LIDS).

The ASP module will be completely revised during the summer of 1970. This revision will both clean up the program and incorporate user feedback resulting from several months experience with the existing ASP module. Complete documentation will be provided for the revised module. In the remainder of this document only the brief User Information section is given.

JOSHUA DPSTM-500	Title Module ASP	Vol. IV	Page C.1.4
		Rev.	Date 6/70

4. User Information

4.1 Input

ASP can be used as a stand-alone module, but it is normally executed as the first module called by RAHAB. The control records input to ASP are:

Name: 'INPUT'. 'ASP'

List: DLAT, VLAT, XVER, TDSN, NCASE, LPRNT

DLAT - Name of this lattice (R*8)
VLAT - Lattice version name (R*8)
XVER - Name of the version of the MULTIGRP Data Set
 to be used (R*8)
TDSN - Temporary Data Set Name (R*8) = 'TEMP'
NCASE - Case number
LPRNT - Print option, 0 = no print, 1 = dump,
 2 = neat print

Name: 'INPUT'. 'RAHAB'. 'GEØM'. ?DLAT. ?VLAT

List: PATRN, (DCEL(N), VCEL(N), N=1,7),
 (NCLA(N), REGNEW(N), (REGØLD(MIN), M=1,4), N=1,8),
 (NCLB(N), REG(N), NSUB(N), NAZM(N), N=1,24),
 (NCLC(N), ISA(N), NZN(N), N=1,24)

PATRN - Name of the lattice pattern (R*8)
DCEL - Cell name (R*8)
VCEL - Cell version name (R*8)
NCLA - Cell number containing REGØLD
REGNEW - New name (R*8) for a homogenous region
 consisting of regions whose names are REGØLD.
REGØLD - Names (R*8) of a group of contiguous regions to
 be homogenized into one region.
NCLB - Cell number containing REG
REG - Name (R*8) of a region to be azimuthally sub-
 divided.
NSUB - Number of azimuthal sub-divisions for region REG.
NAZM - Number of azimuthal sub-divisions into which
 this sub-assembly is to be divided.
NCLC - Cell number containing sub-assembly.

ISA - Sub-assembly number relative to cell NCLC.
NZN - Zone number assigned to this sub-assembly.

Based on the names given in these records, ASP reads lattice pattern data from the LATTICE data set, isotope names from the MULTIGRP data, and cell descriptions from the ASSEMBLY data set.

4.2 Output

The CARD and NAME (See Manuscript B.2) blocks are written on the LIDS.

4.3 Core Size and Timing

The ASP module requires 180K of core, 1 job data set block, and should be executed with a GØTIME=1 minute. Individual sub-program sizes and timings are given in Table I. The timings are for a mixed lattice, 8 isotopes, 8 materials, 6 sub-assemblies and 33 regions.

4.4 Error Stops

All error stops are handled by the JOSHUA Operating System and are classified in five categories. The format of the message is

4-LETTER RØUTINE NAME....MESSAGE NUMBER....MESSAGE...
8-CHARACTERS

4-LETTER RØUTINE NAME - identifies the routine in which
this error occurred.

MESSAGE NUMBER

- the first digit denotes the category of the message (for error routine use). The other digits assist the programmer in pinpointing the error. The following are the five categories:

1. Problem in MANAGE (data overflow).
2. Limit allowed has been exceeded.
3. Important record missing.
4. Material problem in MATER.
5. Missing names (name of region, material or isotope).

MESSAGE

- one of the five described above

8-CHARACTERS

- normally is the region, isotope, or material name being processed when the error occurred.

Table I

Size and Timing for ASP Subprograms
(Size in bytes, time in CPU seconds)

<u>Subprogram</u>	<u>Size</u>	<u>Total Time</u>	<u>Average Time</u>	<u>Entries</u>	<u>Overlay</u>
MAIN	970	.0499	.04992	1	RØØT(1)
PATISO	2944	.3494	.3494	1	RØØT(1)
ERRØR	868	-	-	0	RØØT(1)
DRØPSY	710	-	-	0	RØØT(1)
MANAGE	13378 ⁽¹⁾	1.59744 ⁽²⁾	.08407	19	RØØT(1)
/DATA/	12000	-	-	0	RØØT(1)
/CARD/	7204	-	-	0	RØØT(1)
/TEMP/	7816	-	-	0	RØØT(1)
MAINE	10670	.53248	.53248	1	A(2)
FIND	534	.0832	.00037	225	A(2)
RLR4	430	-	-	0	A(2)
RLL8	444	.11648	.00041	281	A(2)
FIND4	522	-	-	0	A(2)
LEFT	450	.56576	.001	565	A(2)
PATTRN	872	0.0	0.0	3	A(2)
LEFT4	434	0.0	0.0	1	A(2)
RLR8	438	-	-	0	A(2)
RLL4	436	.03328	.00024	139	A(2)
/SCRACH/	7216	-	-	0	A(2)
AGEØM	2142	.01664	.00555	3	B(3)
CGEØM	2266	-	-	0	B(4)
HGEØM	1258	.01664	.00555	3	B(5)
SGEØM	1728	-	-	0	B(6)
DENSY	1634	-	-	0	B(7)
GISMØ	894	0.0	0.0	16	B(7)
CØMCØN	3438	0.0	0.0	8	B(7)
MATER	6748	.46592	.15531	3	B(7)
UAL	690	-	-	0	B(7)
JØS	82420 ⁽⁴⁾				
Totals	164160 ⁽⁵⁾ (ØVLY)	3.8272 ⁽³⁾	.000933	1269	

Footnotes for Table I

- (1) Includes all routines of data manager.
- (2) The CHECK routine uses 1.3 seconds on one entry (output routine).
- (3) Includes time for timing and output routine.
- (4) Includes JØS, other system routines such as \$SEGTAB and timing routine supplied by precompiler and buffers.
- (5) The total size is derived by summing the maximum size overlay level.

Work Done By

W. H. Hightower
H. C. Honeck

Report Written By

W. H. Hightower
H. C. Honeck

Contents

	<u>Page</u>
1. Theory	C.2.3
1.1 Introduction	C.2.3
1.2 Input Description	C.2.3
1.3 Processing Steps	C.2.4
1.4 Region Sub-division	C.2.5
1.5 Boundary Conditions	C.2.6
2. Program Logic	C.2.9
2.1 Interfaces	C.2.9
2.2 Module Structure	C.2.9
3. Coding Details	C.2.11
3.1 COMMON/TEMP/	C.2.11
3.2 Assignment Routines	C.2.12
3.3 Subroutine RCD	C.2.17
4. User Information	C.2.18
4.1 Input	C.2.18
4.2 Output	C.2.18
4.3 Core Size and Timing	C.2.18
4.4 Error Stops	C.2.18

JOSHUA DPSTM-500	Title Module LAGER	Vol. IV	Page C.2.2
		Rev.	Date 5/70

(Blank)

1. Theory

1.1 Introduction

The Lattice GEometry Routine (LAGER) prepares detailed tables describing lattice geometry and composition from brief user-oriented descriptions. For example, the user would describe a cluster in terms of rods, rod centers, and coolant subdivision. LAGER would translate this description into one involving only the basic geometry elements of vertices, lines, and radii needed by a Monte Carlo tracking routine. During this translation, geometric regions are divided into sub-regions, sub-assemblies are collected into zones, and cells are collected into super-cells.

1.2 Input Description

The input to LAGER is the NAME and CARD blocks which are described in manuscript B.2. The CARD block contains both geometry and composition data in a structure resembling a deck of cards. Historically, LAGER was written to operate from cards and then later changed to read the LIDS. The module CLASP can be used to create the NAME and CARD blocks from card input.

Certain features of the input data should be mentioned.

1. Material and geometry data is given separately for each cell.
2. The geometry is described in terms of physical regions.
3. The geometry is measured by dimensions appropriate to individual configurations.

Configuration

Dimension

Annular

Radius

Square or hexagonal

Half-pitch

Cluster

Rod radii and
center position.

4. Elements of the material or geometry are numbered relative to their position in the next larger element, e.g., the first isotope in a material, the first material in a cell, the first region in a sub-assembly, the first sub-assembly in a cell.
5. Boundary conditions are expressed in terms of relative face numbers of adjacent cells.

1.3 Processing Steps

The objective of LAGER is to translate these data into compact blocks which describe a supercell in elementary geometric terms. The following steps are required:

1. Collect all composition data, eliminate duplicate isotope names, and renumber materials and isotopes. The resulting data is written as the COMP block on the LIDS.
2. Divide (as specified in the input) physical regions into sub-regions.
3. Reduce the geometrical configuration to the simple elements of vertices, lines, and radial arcs.
4. Collect all geometry data, renumber all elements, and organize into the GEOT block.

5. Re-express boundary conditions in terms of coordinate transformations.
6. Collect selected geometry elements by zone and organize into the ZONE block.
7. Abstract selected geometry data from the GEOT block and organize into the GEOM block.

The logic required to perform steps 1, 3, 4, and 7 is straightforward and is implied by the structure of the CARD, COMP, GEOT, and GEOM blocks. Steps 2, 5, and 6 need a few additional remarks.

1.4 Region Sub-division

Three types of sub-division specification are given in the CARD block. The first is denoted by ISUB and is the number of radial (or half-pitch) sub-regions into which a region is to be divided. If R_1 and R_2 are the inner and outer radii of an annular region (or the inner and outer half-pitches of a hexagonal or square region), the thickness $R_2 - R_1$ is divided into ISUB equal parts, ΔR . The outer radii of the sub-regions are $R_1 + \Delta R$, $R_1 + 2\Delta R$, etc. This type of sub-division can be used for sub-assembly types NSAK = 2, 3, 4, 5, 8, and 9.

The second type of sub-division specification is denoted by NAZM and is the number of equal azimuthal intervals used to sub-divide a sub-assembly (the entire sub-assembly is sub-divided if NAZM > 1). This can be used for NSAK = 3, 5, and 9. If θ is an angle measured from the positive x axis, and $\theta = 90^\circ$ is the sub-division origin, the angular bounds of the intervals are $\theta_0, \theta_0 + \Delta\theta, \theta_0 + 2\Delta\theta$, etc., where $\Delta\theta = 360/\text{NAZM}$. Note that if a value for ISUB is given for a region, and a value of NAZM is given for the sub-assembly, the region is divided into ISUB*NAZM sub-regions.

The third type of sub-division specification is denoted by IOPTN and is used only for regular cluster geometry (NSAK=6). In a regular cluster the rod centers lie on a set of rings and are uniformly spaced on a ring. The rods on a ring need not, however, be of the same size or contain the same material. It is necessary that adjacent rings contain small enough rods or be sufficiently well separated so that a circle can be drawn through the coolant between adjacent rings without intersecting any rod. If R_n is the radius of rod centers in ring n, and r_n is the radius of the largest rod in ring n, the coolant sub-division circle radius is

$$\rho_n = \frac{1}{2} [(R_n - r_n) + (R_{n+1} - r_{n+1})] \quad (1)$$

The set of circles ρ_n sub-divides the coolant into annular rings pierced by the rods. These annular rings can be azimuthally sub-divided by constructing radial lines between the circles. If $\theta_{k,n}$ is the azimuthal angle of the k^{th} rod in the n^{th} ring, the coolant sub-division radial line is given by

$$\psi_{k,n} = \frac{1}{2} [\theta_{k,n} + \theta_{k+1,n}] \quad (2)$$

The coolant sub-region surrounding the k^{th} rod in the n^{th} ring is then defined by

$$\rho_{n-1} \leq r \leq \rho_n, \quad \psi_{k-1,n} \leq \theta \leq \psi_{k,n} \quad (3)$$

The input option IØPTN controls this coolant sub-division.

<u>IØPTN</u>	<u>Description</u>
-1	No coolant sub-division.
0	Sub-divide the coolant with circles between all rod rings.
1	Same as IØPTN=0 but, in addition, azimuthally sub-divide the outermost coolant ring.
2	Same as IØPTN=1 but azimuthally sub-divide the outermost two coolant rings.
3	etc.

1.5 Boundary Conditions

Following the preliminary processing of boundary conditions according to the formulas and conventions of section 3.2 - 3.3 of the manuscript on "Lattice Geometry", the boundary conditions are given in terms of the displacement factors Δx , Δy , $\Delta \theta$. These factors are given in the GEØT block for each face and for each lattice position for the cell containing this face. The displacement factors are used by Monte Carlo tracking routines in the following manner. When a particle leaves a cell in a given lattice position through a face at the point (x_1, y_1) traveling in a direction θ_1 , the particle enters another cell in another lattice position through another face at the point (x_2, y_2) and is travelling in direction θ_2 . The change in

coordinates and direction is a result of locating the coordinate origin and orientation in the cell containing the particle. The coordinates are related by

$$\begin{aligned}x_2 &= (x_1 + \Delta x) \cos \Delta\theta - (y_1 + \Delta y) \sin \Delta\theta \\y_2 &= (y_1 + \Delta y) \cos \Delta\theta + (x_1 + \Delta x) \sin \Delta\theta \\\theta_2 &= \theta_1 + \Delta\theta\end{aligned}\tag{4}$$

We must next determine the adjacent face numbers to complete the GEOT block. This is done in the following manner.

1. Select a face of a cell in a lattice position and obtain the displacement factors and adjacent lattice position number.
2. Use equation (4) to translate the end points (vertices) of the face.
3. Select another face which is in the proper adjacent lattice position.
4. Test the faces for adjacency, that is, one face lies wholly or partly on the other face.
5. If they are adjacent, enter the adjacent face number in the GEOT block.
6. Repeat from step 3 until all possible adjacent faces have been tried.
7. Repeat from step 1 until all faces and lattice positions have been done.

The results of the identification of adjacent faces are also used to determine the fraction $F(\text{NFC} \leftarrow \text{NAFC})$ of the neutron current density leaving an adjacent face (NAFC) which enters this face (NFC). If the cell containing NAFC can be in LPX different lattice positions, the current density is split into LPX parts of which one enters face NFC. If the area AAFC of face NAFC is larger than the area AFC of face NFC, the current density into NFC is the same as one of the LPX parts leaving NAFC.

$$F(\text{NFC} \leftarrow \text{NAFC}) = 1/\text{LPX} \quad \text{AAFC} \geq \text{AFC} \tag{5}$$

JOSHUA DPSTM-500	Title Module LAGER	Vol. IV	Page C.2.8
		Rev.	Date 5/70

If AFC is greater than AAFC the current density into NFC for each lattice position comes from several NAFC, so that

$$F(NFC \leftarrow NAFC) = AAFC/AFC/LPX \quad , \quad AAFC < AFC \quad (6)$$

The current density fraction is computed from these rules for each NFC and NAFC and stored in the ZONE block.

2. Program Logic

2.1 Interfaces

LAGER reads the following records:

('TEMP'. 'LIDS'.DLAT.VLAT→V)

'INPUT'. 'LAGER'

V. 'NAME'.NCASE.O.O

V. 'CARD'.NCASE.O.O

LAGER writes the following records:

V. 'GEØT'.NCASE.O.O

V. 'GEØM'.NCASE.O.O

V. 'CØMP'.NCASE.O.O

V. 'ZØNE'.NCASE.O.O

2.2 Module Structure

The LAGER module structure can be summarized as follows:

LAGER	(main program)
LCD	(load composition data)
LGD	(load geometry data)
ZTABLE	(prepare ZØNE block)
CGT	(prepare GEØM block)

Subroutine LGD can be further expanded:

LGD	(load geometry data)
LGD1	(arbitrary geometry)
LGD2	(annular geometry)
LGD3	(azimuthally sub-divided annular geometry)
LGD4	(hexagonal geometry)
LGD5	(azimuthally sub-divided hexagonal geometry)
LGD6	(cluster)
LGD8	(square geometry)
LGD9	(azimuthally sub-divided square geometry)

These subroutines correspond to NSAK=1-9 respectively. Each routine takes its input from the CARD block using the RCD subroutine and places its output in the GEOT block. A group of subroutines are used to facilitate assignment of data to the GEOT block.

ASGNCL	(assign a cell)
ASGNSA	(assign a sub-assembly)
ASGNRG	(assign a region)
ASGNSR	(assign a sub-region)
ASGNFC	(assign a face)
ASGNSD	(assign a side)
ASGNRD	(assign a radius)
ASGNVR	(assign a vertex)
FNDSR	(find a sub-region)
FNDVR	(find a vertex)
SDCNT	(count sides)
MØVEFC	(move face)
ADADJF	(add adjacent face)
SYMTRY	(determine symmetry angle)

3. Coding Details

3.1 CØMMØN/TEMP/

The CØMMØN/TEMP/ area is used primarily to pass data between the various geometry loading routines. It is defined by:

CØMMØN/TEMP/KX,KXX,KCØRG,KGØRG,KZØRG,KXØRG,
 NIN,NØUT,NZN,NAZM,AZMØRG,KARD,K,KR,
 NCL,NCLX,NCLXX,KCL,KCLT(7),
 NSA,NSAX,NSAXX,KSA,KSAT(15),
 NRG,NRGX,NRGXX,KRG,KRGT(100),
 NSR,NSRX,NSRXX,KSR,KSRT(100),
 NFC,NFCX,NFCXX,KFC,KFCT(100),
 NSD,NSDX,NSDXX,KSD,KSDT(200),
 NRD,NRDXX,NRDXX,KRD,KRDT(100),
 NVR,NVRX,NVRXX,KVR,KVRT(200),
 METH(15)

Before defining these symbols in detail, some general remarks are necessary. The LGD type routines take data from the CARD block, reformat them, and move them into the GEØT block. As each element of the geometry is processed, it is placed in the next available location in the GEØT block and then linked by pointers to associated elements previously stored. The leading character K on a variable name indicates that it is a pointer. Since several blocks are packed into the DATA array, there are both absolute and relative pointers. If KGØRG is the origin (first location - 1) of the GEØT block, and KR is a relative pointer (the KRth location in the GEØT block) the absolute pointer K (the Kth location in the DATA block) is $K = KR + KGØRG$. Since the DATA and GEØT arrays are equivalent, $DATA(K) = GEØT(KR + KGØRG)$.

With these conventions in mind we define the symbols in the CØMMØN/TEMP/ area.

- KX - Last location of the DATA array used.
- KXX - Last location of the DATA array available for use (dimensioned size of the DATA array).
- KXØRG - Origin (first location - 1) of block X in the DATA array. KCØRG,KGØRG,KZØRG,KXØRG refer to blocks CØMP,GEØT,ZØNE, and XSEC.

JOSHUA DPSTM-500	Title Module LAGER	Vol. IV	Page C.2.12
		Rev.	Date 5/70

NIN - Card input unit number.
 NØUT - Print output unit number.
 NZN - Current zone number.
 NAZM - Current number of azimuthal sub-divisions.
 AZMØRG - Azimuthal sub-division origin ($=90^\circ$).
 KARD - Pointer to a group of 6 values in the array
 CARD(I,KARD), I=1,6.
 K - Next available location in the DATA array for
 storing data in the GEØT block.
 KR - K-KGØRG.
 METH - Collision probability method to be used for
 sub-assembly NSA is METH(NSA).
 NCL - Last cell number assigned.
 NCLX - Number of cells in the supercell.
 NCLXX - Maximum number of cells allowed.
 KCL - Absolute pointer to last cell assigned.
 KCLT - Table of absolute pointers to cells by cell
 number, NCL.

3.2 Assignment Routines

All assignment routines are used by an LGD type routine to accomplish a specific task. Example: ASGNCL adds cell data to the GEØT block. A brief discussion of the functions performed by each routine follows. All data are added sequentially and are actually placed starting at the next available location (at K).

ASGNCL (ISAX,KCLNAM,ILPX)

1. Increment cell counter (NCL).
2. Store forward pointer in either the last assigned cell area or the header area if this is the first cell area assigned. (KCL points to this location).
3. Store number of sub-assemblies in this cell (ISAX), cell name pointer (KCLNAM) and number of lattice positions that this cell is in (ILPX).
4. Search all previously assigned sub-assembly areas for the appearance of this cell number. If found, store the pointer to this cell.
5. Assume that the sub-assembly numbers in this cell will be NSA+I where I=1,ISAX, and store these values. Pointers to these sub-assemblies have not yet been assigned.

JOSHUA DPSTM-500	Title Module LAGER	Vol. IV	Page C.2.13
		Rev.	Date 5/70

6. Update K, KR, and KCL.

ASGNSA (NCLP,NSAK,NZNP,IRGX)

1. Increment sub-assembly counter (NSA).
2. Store forward pointer in either the last assigned sub-assembly area or the header area.
3. Store cell (NCLP) and zone (NZNP) numbers to which this sub-assembly belongs, sub-assembly kind (NSAK), the number of regions in this sub-assembly (IRGX), and the number of azimuthal sub-divisions (NAZM).
4. Search last cell area assigned for the appearance of this sub-assembly number. If found, store pointer to this sub-assembly.
5. Search all region areas assigned for this sub-assembly number. If found, store pointer.
6. Assume that region numbers in this sub-assembly will be NRG+I where I=1, IRGX. Pointers are not stored at this time.
7. Update K, KR and KSA.

ASGNRG (NSAP,VRG,ISRX,KRGNAM)

1. Increment region counter (NRG).
2. Store forward pointer in either the last assigned region or header area.
3. Store sub-assembly number (NSAP) to which this region belongs, volume of the region (VRG), name pointer (KRGNAM), and the number of sub-regions in this region (ISRX),
4. Search last sub-assembly assigned for this region number. If found, assign pointer.
5. Search all sub-region areas assigned for this region number. If found, assign region pointer.
6. Assume that sub-region numbers in this region will be ISR+I where I = 1, ISAX. Pointers are not stored at this time.
7. Update K, KR and KRG.

ASGNSR (NRGP,VSR,ISDX,KMT)

1. Increment sub-region counter (NSR).
2. Store forward pointer in either the last assigned sub-region or header area.
3. Store region number (NRGP) to which this sub-region belongs, volume of sub-region (VSR), pointer (KMT) to the material in composition table, and the number of sides surrounding this region (ISDX).
4. Search last region assigned for this sub-region number. If found, store pointer to this sub-region.
5. Search all face and side blocks for this sub-region number. If found, store pointer to this sub-region.
6. Update K, KSR, KR.
7. No side numbers or pointers are stored. However, space is allocated for them.

ASGNSD (NSR1,NSR2,NFCP,NRDP,NVR1,NVR2)

1. Increment side counter (NSD).
2. Store forward pointer in either the last side assigned or the header.
3. Store sub-region numbers (NSR1,NSR2) to which this side belongs (if either sub-region is not known then zero is stored), the face number (NFCP), the radius number (NRDP) and vertex numbers (NVR1,NVR2) associated with this side.
4. Store pointers to side data if known.
5. Search sub-region and face areas for side number. If found, store pointer.
6. Update K, KSD, KR.

ASGNRD(RD,XR,YR)

1. Increment radius counter (NRD).
2. Store forward pointer in either the last radius assigned or the header.
3. Store the radius (RD) and x-y coordinates (XR,YR) relative to the center of this cell.
4. Search all side blocks for this radius number. If found, store pointer.
5. Update K, KR and KRD.

ASGNVR (XV,YV)

1. Increment vertex counter (NVR).
2. Store forward pointer in either the last vertex area assigned or the header.
3. Store the x-y coordinates (XV,YV) of the vertex relative to the center of the cell.
4. Search all side areas for this vertex number. If found, assign vertex pointer.
5. Update K, KR, KVR.

ASGNFC (NSRP,NSDP,AFC,IØFT,ILPX,MLP,XDIS,YDIS,RØT,MALP)

1. Increment face counter (NFC).
2. Store forward pointer in either the last face block assigned or the header.
3. Store sub-region number (NSRP), side number (NSDP), face area (AFC), number of lattice positions (ILPX). For each lattice position store lattice position number (MLP) and pointer, x and y displacement (XDIS,YDIS), rotation angle (RØT), and adjacent lattice position number (MALP).
4. Space is left for one adjacent face. After all sub-assemblies are processed the sub-assemblies are connected by way of faces. If more than one face is found to be adjacent to another face then that face will be moved using subroutine MOVEFC and expanded using subroutine ADADJF.

5. If the side associated with this face has been assigned, the side's sub-region NSR1 or NSR2 is zeroed out if IØFT is 1 or 0 respectively. IØFT=0 implies that this is an exterior face, i.e., there is no sub-region beyond this face of this sub-region starting from the center of this sub-assembly moving toward the face in question. The converse is true for IØFT=1. The sides, face and pointer are stored.
6. Search all face areas for this face and store pointer.
7. Update K, KR and KFC.

The following routines are used by LGD while setting boundary conditions (connecting faces):

MØVEFC(KLP,NFCP,IMØVE)

Move the data for face number NFCP to the end of the GEØT block. Update the lattice position pointer (KLP) and move counter (IMØVE).

ADADJF(KLPP,KFCPP,NFCP)

Add adjacent face number NFCP to a face previously moved to the end of the GEØT block at KFCPP.

The following routines are used by LGD6:

SDENT(SANGL,EANGL,ANGLS,NØSF,NØSD)

Count the sides on a radial ring (ANCIS) and return that count (NØSF) and relative side numbers (NØSD).

FNDVVR(DTH1,DTH2,FROM,NVR1,NVR2)

Starting from angle FRØM search vertex tables DTH1 and DTH2 for vertices on the same angle and return their relative vertex numbers NVR1 and NVR2 respectively.

FNDSR(NSDP,NSR1,NSR2)

This routine searches all sub-region areas for side NSDP. The sub-region's number is stored in NSR1 and NSR2 if a second sub-region is found.

SYMTRY(ISMTRY,NRDPØS,RØDPØS,NRGTYF,KRØD,EPS,KFP,ICENTX,KMTT,RØDRAD)

The symmetry angle of the cluster is determined and is returned as 360/(symmetry angle) in ISMTRY. ISMTRY is stored in the sub-assembly area by LGD6.

3.3 Subroutine RCD

The subroutine RCD fetches a string of six numbers from the CARD block. The CARD block is an array

(CARD(N,KARD),N=1,6),KARD=1,NCARD

and simulates a deck of NCARD cards with 6 fields/card. Data in the CARD array may be either an integer (I), a floating point number (F), or not used (Z). Some of the integers point to 8-character names in the NAME block. There are 17 different arrangements of the six fields.

<u>ITYPE</u>	<u>FIELD</u>					
	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>
1	I	Z	Z	Z	Z	Z
2	I	I	I	Z	Z	Z
3	I	I	I	I	F	F
4	I	I	Z	Z	Z	Z
5	I	I	I	I	Z	Z
6	I	I	I	I	I	Z
7	I	I	I	I	I	I
8	I	I	I	I	Z	Z
9	I	I	F	I	I	Z
10	I	F	Z	Z	Z	Z
11	I	I	I	F	I	Z
12	I	I	I	F	I	F
13	I	I	I	I	F	I
14	I	I	F	F	Z	Z
15	I	I	I	F	F	F
16	I	F	F	F	Z	Z
17	I	F	F	Z	Z	Z

Subroutine RCD is used with the statement:

CALL RCD(AR1,AR2,AR3,AR4,AR5,AR6,ITYPE)

Based on the value of ITYPE, the subroutine will return the integer or floating point values of CARD(N,KARD),N=1,6 in the six arguments AR1 - AR6. The value of KARD is in COMMON/TEMP/ and is automatically advanced by one each time RCD is called. The programmer should use symbols for AR1 - AR6 whose types (I*4 or R*4) are consistent with those specified by the value of ITYPE.

The five categories are:

1. Relative number of sequence is out of range.
2. Improperly calculated or out of tolerance value.
3. Could not be found (some value on name).
4. Storage overflow (some array is full).
5. Number exceeds the maximum allowed.

TABLE I

Size and Timing for LAGER Subprograms

(Size in bytes, time in CPU seconds)

<u>Subprogram</u>	<u>Size</u>	<u>Total Time</u>	<u>Average Time</u>	<u>Entries</u>	<u>Overlay</u>
MAIN	2,478	.09984	.09984	1	RØØT(1)
RCD	614	.03328	.00026	127	RØØT(1)
MANAGE	24,024 ¹	2.09664 ²	.027229	77	RØØT(1)
LEFT	450	0.0	0.0	3	RØØT(1)
ERRØR	1,038	-	-	0	RØØT(1)
/CARD/	7,204	-	-	0	RØØT(1)
/DATA/	20,000	-	-	0	RØØT(1)
/TEMP/	4,332	-	-	0	RØØT(1)
/SCRACH/	3,360	-	-	0	RØØT(1)
LCD	1,912	0.0	0.0		A(2)
LGD	5,134	.38272	.38272	1	A(3)
ASANCL	916	0.0	0.0	3	A(3)
ASANSA	1,110	0.0	0.0	6	A(3)
ASGNRG	1,102	.06656	.00202	33	A(3)
ASGNSR	1,164	.11648	.00265	44	A(3)
ASGNSD	1,392	.11648	.00188	62	A(3)
ASGNRD	686	.03328	.00071	47	A(3)
ASGNVR	716	.01664	.00092	18	A(3)
ASGNFC	1,654	.06656	.00277	24	A(3)
ADADJF	692	0.0	0.0	48	A(3)
MØVEFC	994	-	-	0	A(3)
LGD1	4,064	-	-	0	B(4)
LGD2	2,066	.03328	.01109	3	B(5)
LGD3	3,324	-	-	0	B(6)
LGD4	3,488	.8320	.02773	3	B(7)
LGD5	4,782	-	-	0	B(8)
LGD6	12,336	-	-	0	B(9)
SYMTRY	2,040	-	-	0	B(9)
SDCNT	1,094	-	-	0	B(9)
FNDVR	938	-	-	0	B(9)
FNDSR	792	-	-	0	B(9)

TABLE I (Cont'd.)

Size and Timing for LAGER Subprograms

(Size in bytes, time in CPU seconds)

<u>Subprogram</u>	<u>Size</u>	<u>Total Time</u>	<u>Average Time</u>	<u>Entries</u>	<u>Overlay</u>
LGD8	3,404	-	-	0	B(10)
LGD9	4,764	-	-	0	B(11)
ZTABLE	2,544	.06656	.06656	1	A(12)
CGT	2,316	.03328	.03328	1	A(13)
JØS	<u>83,200³</u>	<u>-</u>	<u>-</u>	<u>0</u>	RØØT(1)
TOTALS	179,460 ⁴ (ØVLY)	3.2440 ²	.00645	503	

Footnotes for Table I.

¹Includes all routines of data manager.

²Includes time for JØSHUA Operating System and output routines.

³Includes JØSHUA Operating System and buffers.

⁴Total size is the sum of the overlay sizes including JØS, buffers, precompiler-generated timing routines, and system routines such as \$SEGTAB.

Work Done By

W. H. Hightower

Report Written By

W. H. Hightower

CONTENTS

	<u>Page</u>
1. Theory	C.3.3
2. Program Logic (not included)	
3. Coding Details (not included)	
4. User Information	C.3.4
4.1 Input	C.3.4
4.2 Output	C.3.4
4.3 Error stops	C.3.4

JOSHUA DPSTM-500	Title Module CLASP	Vol. IV	Page C.3.2
		Rev.	Date 6/70

(Blank)

1. Theory

For most problems, users create material and geometry descriptions of cells on the ASSEMBLY data set and use the ASP module to process these data into the NAME and CARD blocks on the Lattice Intermediate Data Set (LIDS). A Card Loading version of ASP (called CLASP) has been written for on-site users without convenient access to the JOSHUA terminals, and for off-site users. This module reads a deck of cards structured identically to the CARD block except that the cards do contain 8 character names. CLASP reads the cards, moves all names into the NAME block, replaces the names with pointers, and writes the NAME and CARD blocks.

JOSHUA DPSTM-500	Title Module CLASP	Vol. IV	Page C.3.4
		Rev.	Date 6/70

4. User Information

4.1 Input

CLASP can be executed as a stand-alone module but it is normally executed by the control module RAHAB. In either case the required input is:

1. A deck of cards structured identically to the CARD block described in manuscript B.2, section 17.

2. The record

Name: 'INPUT'. 'CLASP'

List: DLAT,VLAT,TDSN,NCASE,LPRNT

DLAT - Lattice name (R*8)

VLAT - Lattice version (R*8)

TDSN - Temporary data set name = 'TEMP' (R*8)

NCASE - Case number

LPRNT - Print option

= 0, not print

= 1, dump

= 2, neat print

4.2 Output

The CARD and NAME blocks are written on the LIDS.

4.3 Error Stops

All error stops consist of a brief message and a JOSHUA Operating System message. Both messages are self-explanatory.

JOSHUA
DPSTM-500

Title Part D - Nuclear Data Modules

Vol. IV	Page D.O.1
Rev.	Date 6/70

Contents

- D.1 Cross Section Input Processor - Module CRISP
- D.2 Resonance Data Input - Module RESIN
- D.3 Resonance Calculation - Module CREEP

JOSHUA DPSTM-500	Title Part D - Nuclear Data Modules	Vol. IV	Page D.O.2
		Rev.	Date 6/70

(Blank)

Work Done By

C. E. Henderson
H. C. Honeck

Report Written By

C. E. Henderson
H. C. Honeck

Contents

	<u>Page</u>
1. Theory	D.1.3
1.1 Purpose	D.1.3
1.2 Energy Bands	D.1.3
1.3 Data Packing	D.1.3
2. Program Logic	D.1.4
2.1 Interfaces	D.1.4
2.2 Over-all Structure	D.1.4
2.3 Data Input Logic	D.1.4
2.4 Calculation of the XSEC Block Size	D.1.7
2.5 Calculation of the CHI Block Size	D.1.9
2.6 Calculation of the SMAT Block Size	D.1.11
2.7 Calculation of the TMAT Block Size	D.1.13
2.8 Energy Band Structure Calculation	D.1.13
2.9 Preparation of XSEC and CHI Blocks	D.1.15
2.10 Preparation of SMAT Blocks	D.1.16
2.11 Size Provisions	D.1.16
3. Coding Details	D.1.17
3.1 COMMON/DATA/	D.1.17
3.2 COMMON/SPECS/	D.1.17
3.3 COMMON/GRØUPS/	D.1.18
3.4 COMMON/TEMP/	D.1.18
3.5 COMMON/SIZE/	D.1.19
3.6 COMMON/SCRACH/	D.1.19
3.7 Main Program CRISP	D.1.19
3.8 Subroutine LGRPS	D.1.20
3.9 Subroutine SIZE1	D.1.20
3.10 Subroutine SIZE2	D.1.20
3.11 Subroutine SIZTM	D.1.20
3.12 Function LXSEC	D.1.21
3.13 Function LSMAT	D.1.21
3.14 Subroutine CALEB	D.1.21
3.15 Subroutine LMGXS	D.1.21
3.16 Subroutine LMGSM	D.1.21
4. User Information	D.1.22
4.1 Input	D.1.22
4.2 Output	D.1.22
4.3 Core Size and Timing	
4.4 Error Stops	

JOSHUA DPSTM-500	Title Module CRISP	Vol. IV	Page D.1.2
		Rev.	Date 6/70

(Blank)

1. Theory

1.1 Purpose

The purpose of the CRISP module is to compute the energy band structure and to organize the cross section and scattering matrix data (Cross Interaction, Selection, and Processing).

1.2 Energy Bands

The FIST module has the most severe core storage requirements of all the modules called by RAHAB. In most cases all of the required data cannot be held in core so that the data must be segmented and shuffled in and out of core. The most efficient way to segment the data is to collect energy groups into "energy bands". For example, in the HAMMER type 84 group structure, one might select three bands containing groups 1 - 30, 31 - 54, and 55 - 84. The segmentation into bands should be done at execution time since the FIST storage requirements depend on the spatial description and number of isotopes as well as the number of groups.

The segmentation into energy bands must recognize that there are two natural places for the band limits and that excessive FIST running time will result from a poorly chosen band structure. The most important region is the thermal region since it requires the largest number of outer iterations. The thermal data should be placed in as few bands as possible. The next most important region is the fast region where the fission spectrum is non-zero. Several outer iterations are generally required in the fast region; so that, again, the fast data should be placed in as few bands as possible.

1.3 Data Packing

Segmentation of the data into energy bands requires that FIST shuffle data blocks in and out of core. If a problem has 20 isotopes and an average of 30 groups/band, there may be as many as 620 read requests to the data manager to obtain the MULTIGRP data for one band. The number of requests can be dramatically reduced if the data from MULTIGRP is first organized into large blocks which are then written on the Lattice Intermediate Data Set (LIDS). The LIDS blocks XSEC, CHI, and SMAT are used for this purpose and are principal outputs from CRISP.

2. Program Logic

2.1 Interfaces

The following records are read by CRISP:

('TEMP'.'LIDS'.DLAT.VLAT→V)

'INPUT'.'CRISP'

V.'NAME'.NCASE.O.O

V.'CØMP'.NCASE.O.O

V.'GEØM'.NCASE.O.O

V.'RESD'.NCASE.O.O

V.'ZØNE'.NCASE.O.O

V.'GEØT'.NCASE.O.O

In addition, the entire MULTIGRP data set is required.

The following records are written by CRISP:

V.'XSEC'.NCASE.NEB.O

V.'SMAT'.NCASE.NEB.O

V.'CHI' .NCASE.O .O

V.'GRPS'.NCASE.O .O

2.2 Over-all Structure

The operation of the module can be divided into four steps as shown in Figure 1. The following sections describe these steps in detail.

2.3 Data Input Logic

1. Read 'INPUT'.'CRISP' and establish record names.
2. Clear storage arrays.
3. Read records NAME, CØMP, GEØM, RESD, ZØNE, and compute their size.
4. Read group structure from MULTIGRP and compute group widths, $DE(I) = E(I) - E(I+1)$.
5. Read CØNTENTS and LIMITS records from MULTIGRP for each isotope. Store the cross section limits in arrays IHI (NDT,NIS), ILØ (NDT,NIS).

<u>NDT</u>	<u>Data Type</u>	<u>NDT</u>	<u>Data Type</u>	<u>NDT</u>	<u>Data Type</u>
1	Total	6	(n,2n)	11	(n,D)
2	Capture	7	Chi	12	(n,T)
3	Fission	8	(n,v)	13	(n,v*)
4	v	9	(n, α)		
5	Transport	10	(n,p)		

Save JHSX(NIS) from CØNTENTS record. Store scattering matrix limits JLTMT(NIS), JHTMT(NIS), LPTØT(J,NIS), ILTØT(J,NIS), and IHTØT(J,NIS).

Figure 1. CRISP Flow Diagram

2.4 Calculation of XSEC Block Size

The structure of the XSEC is shown in Figure 2 (see manuscript B.2). We will need to calculate the size of these blocks in advance of actually loading the data in order to calculate the energy band structure. Let IEBL and IEBH be the group limits of an energy band. Then the length (in 4-byte words) of the XSEC block for that band is given by:

$$LXSEC = \sum_{NIS} U(NIS) \left\{ 18 + \sum_{NDT=1}^B V(NDT, NIS) [2 + IH - IL] \right\}$$

where $IH = \text{MINO}(IHI(NDT, NIS), IEBH)$

$IL = \text{MAXO}(ILO(NDT, NIS), IEBL)$

and

$V(NDT, NIS) = 1$ if $IHI(NDT, NIS) \geq IEBL$ and
 $ILO(NDT, NIS) \leq IEBH$

$= 0$ otherwise

$U(NIS) = 1$ if any $V(NDT, NIS) = 1$

$= 0$ if all $V(NDT, NIS) = 0$

XSEC(K) KXPT(2,K)		Prologue for Each Isotope	
KXS+0	KNXTXS	PØINTER TØ CRØSS SECTIONS FØR NEXT ISØTOPE	
1	NIS	ISØTOPE NØ. AND PØINTER TØ CØMP BLØCK	
2	KDT1	PØINTER TØ DATA TYPE 1 Total	
3	KDT2	2	Caption
4	KDT3	3	Fission
5	KDT4	4	Nu
6	KDT5	5	Transport
7	KDT6	6	(n,2n)
8	KDT7	7	Not used.
9	KDT8	8	(n,v)
10	KDT9	9	(n,a)
11	KDT10	10	(n,p)
12	KDT11	11	(n,D)
13	KDT12	12	(n,T)
14	KDT13	13	(n,v*)
15	KDT14	14	
16	KDT15	15	
17	KDT16	16	

Cross Section Data		
KDT+0	IA	IB
1	$\sigma(IA)$	
2	$\sigma(IA+1)$	

	$\sigma(IB)$	

Block name: 'XSEC'.NCASE.NEB.0

Figure 2. Schematic of the XSEC Block

2.5 Calculation of the CHI Block Size

Fission spectrum vectors are packed by isotope into the CHI block shown in Figure 3 (see manuscript B.2). The length (in 4-byte words) of the CHI block is given by:

$$LCHI = \sum_{NIS} W(NIS) \left[3 + IHI(7, NIS) - ILØ(7, NIS) \right]$$

where $W(NIS) = 1$ if $IHI(7, NIS) > 0$
 $= 0$ otherwise

JOSHUA DPSTM-500	Title Module CRISP	Vol.	Page
		IV	D.1.10
		Rev.	Date 6/70

CHIT(K)
KCPT(2,K)

KCH+0

NIS	KNXTCH
IA	IB
CHI(IA)	
CHI(IA+1)	

CHI(IB)	

ISOTOPE NUMBER, POINTER TO NEXT ISOTOPE
LOWER AND UPPER GROUP LIMITS

Block name: 'CHI'.NCASE.O.O

Figure 3. Schematic of the CHI Block

2.6 Calculation of the SMAT Block Size

Scattering matrix data are packed by isotope into the SMAT blocks. The description of these blocks is given in Figure 4 (see manuscript B.2). In RAHAB only data type 4 (total) for Legendre number 0 and 1 are used.

The length of the SMAT block (4-byte words) is given by:

$$LSMAT = \sum_{NIS} V(NIS) \left[11 + \sum_{J=IEBL}^{IEBH} LSM(J,NIS) \right]$$

where

$$LSM(J,NIS) = 3 + LP \left[2 + IHT\emptyset T(J,NIS) - ILT\emptyset T(J,NIS) \right]$$

$$\begin{aligned} LP &= LPT\emptyset T(J,NIS) \text{ if } LPT\emptyset T(J,NIS) \leq LPX \\ &= LPX \text{ if } LPT\emptyset T(J,NIS) > LPX \end{aligned}$$

and

$$\begin{aligned} V(NIS) &= 1 \text{ if } JHTMT(NIS) \geq IEBL \text{ and } \\ &\quad JLTMT(NIS) \leq IEBH \\ &= 0 \text{ otherwise} \end{aligned}$$

Figure 4. Schematic of the SMAT Block

SMAT(K)
KSPT(2,K)

KSM+0	IDBF	KNXTSM
1	NIS	KIS
2		KDT1
3		KDT2
4		KDT3
5		KDT4
6		KDT5
7		KDT6
8		KDT7
9		KDT8

Prologue for Each Isotope

IDBF=0 (FULL MATRIX),=1 (HALF), POINTER TO DATA FOR NEXT ISOTOPE
ISOTOPE NO. AND POINTER TO COMP TABLE

POINTER TO DATA TYPE 1 (ELASTIC)
2 (INELASTIC)
3 (n,2n)
4 (TOTAL = EL + IN + 2*N2N)
5
6
7
8

Scattering Matrix Data

KDT	JA	JB
KJ=KDT+1	J	KNXTJ
KJ+1	LX	IX
+2	IA	IB
+3	$\sigma_0(J)$	
	$\sigma_0(IA \rightarrow J)$	
	$\sigma_0(IA+1 \rightarrow J)$	

	$\sigma_0(IB \rightarrow J)$	
	$\sigma_1(J)$	
	$\sigma_1(IA \rightarrow J)$	
	$\sigma_1(IA+1 \rightarrow J)$	

	$\sigma_1(IB \rightarrow J)$	

	$\sigma_{LX}(IB \rightarrow J)$	

LOWER AND UPPER FINAL GROUP LIMITS
FINAL GROUP NO., POINTER* TO NEXT J
MAX. LEGENDRE ORDER GIVEN, $IX=IB-IA+2$
LOWER AND UPPER INITIAL GROUP
SCATTERING MATRIX DATA AND CROSS SECTION

IF $IA \leq I \leq IB$ AND $L \leq LX$

THE CROSS SECTION IS AT:

$$K = KJ + L*IX + 3$$

MATRIX ELEMENTS ARE AT:

$$K = KJ + L*IX + I - IA + 4$$

Block name: 'SMAT' NCASE.NEB.O

2.7 Calculation of the TMAT Block Size

Transport matrices are packed into the TMAT blocks and the length of these blocks is needed to determine the energy band structure. The length (4-byte words) of the TMAT blocks is calculated by the subroutine SIZTM. For details, refer to manuscript E.1 and section 3.11.

2.8 Energy Band Structure Calculation

The DATA array in the FIST module contains the following blocks.

<u>Name</u>	<u>Size</u>	<u>Size Calculation Method</u>
NAME	LNAME	Reading
CØMP	LCØMP	Reading
GEØM	LGEØM	Reading
RESØ	LRESØ	Reading
ZØNE	LZØNE	Reading
FLUX	LFLUX	= NSRX*NEG
JIN	LJIN	= NFCX*NEG
JØUT	LJØUT	= NFCX*NEG
QCEL	LQCEL	= NCLX*NEG
CHI	LCHI	(See previous sections.)
XSEC	LXSEC	(See previous sections.)
SMAT	LSMAT	(See previous sections.)
TMAT	LTMAT	(See previous sections.)

The first 10 blocks are fixed length. The sizes can be obtained by reading the block (NAME through ZØNE), computed from the number of sub-regions, faces, cells, and groups (FLUX through QCEL), or computed from the limits given on MULTIGRP (CHI). The size of the last 3 blocks depend on the energy band structure. The rules for the XSEC, SMAT, and TMAT blocks were given in the previous section.

The energy band structure can be determined in the following manner. First, let LFIST be the length of the DATA array allocated in the FIST module. Subtract from this the length of the 10 fixed length blocks plus the constant amount of LTMAT and determine the storage left.

LEFT=LFIST-LNAME-LCOMP-LGEOM-LRESL-LZONE
-LFLUX-LJIN-LJOUT-LQCEL-LCHI-LTMATA

The problem is to select values of IEBL and IEBH such that

$$\text{LEFT} \geq \text{LTMATB} * (\text{IEBH} - \text{IEBL} + 1) + \text{LXSEC}(\text{IEBL}, \text{IEBH}) \\ + \text{LSMAT}(\text{IEBL}, \text{IEBH})$$

where both LXSEC and LSMAT are shown as functions of the band limits. Define

NEG - number of energy groups
JHCHI - largest J for non-zero fission spectrum
JFTG - first thermal group number
NEB - energy band number
NEBX - number of energy bands
IEBH(N) - highest group number in energy band N

The rules for selecting the energy bands are as follows:

1. Set a flag IDBT=0. If the detailed balance flag is off (IDBF=0), or if there is only one thermal group (NEG-JFTG<1), skip this step. If the flag is on (IDBF=1), calculate LEFT using IEBL=JFTG, IEBH=NEG. If LEFT > 0, omit the remainder of this step. If LEFT < 0, the thermal range will be divided into several bands and detailed balance cannot be used. Set IDBF=0 and IDBT=1. Correct the limits IHTOT(J,NIS) for J=JFTG, NEG-1 as follows:

- a. Set I=NEG
- b. If ILTOT(I,NIS) ≤ J go to step d
- c. Set I=I-1 and if I > J repeat step b
- d. Set IHTOT(N,NIS)=I

2. Establish a tentative energy band structure as follows:

- a. Set IL=IH=1, NEB=0
- b. Calculate LEFT using IL,IH for band limits.
- c. If LEFT < 0 increment NEB and enter IH-1 into IEBH(NEB). An error occurs if a band cannot contain one full group. Set IL=IH and go back to step b.
- d. If LEFT > 0, increment IH and go back to step b if IH ≤ NEG.
- e. Increment NEB and enter NEG into IEBH(NEB). Set NEBX=NEB

3. If $NEBX=1$, or if $IEBH(NEBX-1) < JFTG$, all thermal groups are in the last band. Go to step 7.
4. Locate the band in which $JFTG$ is included, that is, the value of NTB for which $IEBH(NTB) > JFTG > IEBH(NTB-1)$. The number of bands used for the thermal region is $NEBX-NTB+1$.
5. Repeat the procedure in step 2 starting with $IL=IH-JFTG$ and array $IEBH(NEBP)$. The number of bands will be $NEBXP$. If $NEBXP \geq NEBX-NTB+1$, go to step 7.
6. Change the limit $IEBH(NTB)$ to $JFTG-1$. Move the limits $IEBHP(K)$ into $IEBH(K_NTB)$ for $K=1, NEBXP$. Reset $NEBX=NTB+NEBXP$.
7. Store band structure results and write the CRPS record.
8. Save the length of the CHI, XSEC, and SMAT blocks for later use.

2.9 Preparation of XSEC and CHI Blocks

1. Delete all DATA blocks except NAME and COMP.
2. Allocate space for CHI and for each XSEC block.
3. Read cross sections from MULTIGRP by isotope and move data into the CHI and XSEC blocks. Use the limits data in $IHI(NDT, NIS)$ and $ILØ(NDT, NIS)$ to expedite the move. Use the data in $JHSX(NIS)$ to avoid reading omitted secondary cross section records.
4. Write the CHI and XSEC blocks.
5. Delete the CHI and XSEC blocks.

JOSHUA DPSTM-500	Title Module CRISP	Vol. IV	Page D.1.16
		Rev.	Date 6/70

2.10 Preparation of SMAT Blocks

1. Loop over energy bands.
2. Allocate space for the SMAT block of given energy band.
3. Start loop over isotopes.
4. Set up prologue to SMAT block.
5. Start loop over final groups J.
6. Skip to step 7 if $J > JHTMT(NIS)$ or $J < JLTMT(NIS)$.
7. Read in total scattering matrix record from MULTIGRP and move into proper SMAT block, dividing out the group width $DE(J)$.
8. End group loop.
9. End isotope loop.
10. Write SMAT block, deleting the block after it is written. If $IDBT=1$ and if the block contains data from the thermal energy range, the blocks are not written nor deleted.
11. End loop over energy bands.
12. If $IDBT=1$, calculate detailed balance terms and store for $J \geq JFTG$,

$$\Sigma(J \rightarrow I) = \Sigma(I \rightarrow J) * M(I)/M(J)$$

$$M(I) = E(I) * \exp(-E(I)/TEV)$$

$$TEV = 0.86166E-4 * (273.6 + ITDC)$$

$$ITDC \text{ - temperature given in COMP block.}$$
13. If $IDBT = 1$ write blocks containing data from the thermal energy range.

2.11 Size Provisions

Space is provided for 100 groups, 50 isotopes.

3. Coding Details

The CRISP module is divided into the following parts:

1. CØMMØN/DATA/
2. CØMMØN/SPECS/
3. CØMMØN/GRØUPS/
4. CØMMØN/TEMP/
5. CØMMØN/SIZE/
6. CØMMØN/SCRACH/
7. Main program CRISP
8. Subroutine LGRPS
9. Subroutine SIZE1
10. Subroutine SIZE2
11. Subroutine SIZTM
12. Function LXSEC
13. Function LSMAT
14. Subroutine CALEB
15. Subroutine LMGXS
16. Subroutine LMGSM and DØDEBA
17. Subroutine MANAGE (See manuscript B.1.)

3.1 CØMMØN/DATA/

The LIDS blocks NAME, CØMP, GEØM, RESD, ZØNE, XSEC, and SMAT are packed into the array defined by

CØMMØN/DATA/DATA(15000).

For a description of this array and its usage see manuscript B.1.

3.2 CØMMØN/SPECS/

The record 'INPUT'. 'CRISP' contains the data:

TDSN	-	Temporary data set name = 'TEMP'.
DLAT	-	Lattice name.
VLAT	-	Lattice version.
NCASE	-	Case number.
XDSN	-	Cross section data set name.
IDØ(N) = 0,		calculate size of TMAT block for zone N
> 0,		separate TMAT block for zone N is not given.
		Ignore size calculation.

- XVER - Cross section data set version.
- LPRNT - If LPRNT=1, the maximum length of the DATA array used will be printed.
- LPX - Maximum scattering Legendre index.

These data are stored in /SPECS/.

3.3 COMMON/GRUUPS/

This area contains the group structure information as obtained from the MULTIGRP library. The quantities read in or calculated by CRISP, are as follows:

- NEG - Number of energy groups.
- JDTG - Index to first thermal group.
- IDBF - Detailed balance flag
 - 0 - full matrix
 - 1 - half matrix
- E(I) - Energy break points (EV)
- EBAR(I) - Characteristic energy (EV).
- DE(I) - Delta energy or each energy group.
- NEBX - Total number of energy bands.
- IEBH(NEB) - Upper break point for energy band NEB.

3.4 COMMON/TEMP/

This area is used to pass information from the CONTENTS and LIMITS records in MULTIGRP to the subroutines of CRISP. It also passes information such as the number of isotopes, the number of reaction types, and a detailed balance flag. The items in this area are as follows:

- NISX - Number of isotopes.
- NDTX - Number of reaction types.
- IDBT - Detailed balance flag (see Section 2.8).
- KCØRG - Origin of CØMP block.
- JHSX(NIS) - Highest group number for which secondary cross-section data are given. If none, JHSX=0.
- IHI(NDT,NIS)- Highest group index J for non-zero data for reaction type NDT and isotope NIS.

- ILØ(NDT,NIS) - Lowest group index J for non-zero data for reaction type NDT and isotope NIS.
- LPTØT(J,NIS) - Maximum value of LP used for group J for total scattering data.
LP is angular index LP=L+1
L is the Legendre order index
L=0,1,2,...
- ILTØT(J,NIS) - Lower limit of initial energy group index I for isotope NIS and final group index J.
- IHTØT(J,NIS) - Higher limit of initial energy group index I for isotope NIS.
- LSM(J,NIS) - Information used to calculate the size of SMAT block.
- JLTMT(NIS) - Lowest group number for which total transfer matrix data are given. If none, JHTMT(NIS)=0.
- JHTMT(NIS) - Highest group number for which total transfer data are given. If no data, JHTMT(NIS)=0.

3.5 CØMMØN/SIZE/

This area is used to pass certain size quantities to the subroutines used in calculating the energy band structure. The items stored in this area are:

- LFIST - Number of words in the DATA array in module FIST.
- LEFT - The number of words available in the DATA array in FIST for loading the XSEC, SMAT, and TMAT blocks.
- LTMATB - Factor used in determining the length of TMAT for a given energy band.
- LCHI - The length of the CHI block.

3.6 CØMMØN/SCRACH/

This area is used by SIZE1, SIZE2, LXSEC, LSMAT, SIZTM, LMGXS, and LMGSMT as temporary storage for large arrays.

3.7 Main Program CRISP

The main program CRISP reads the input record and the LIMITS and CØNTENTS records from MULTIGRP for all isotopes. If a CØNTENTS record is not available for the isotope at the temperature given in the CØMP block, CRISP assumes the isotope is temperature independent and tries to read a CØNTENTS record at zero temperature. If the read

JOSHUA DPSTM-500	Title Module CRISP	Vol. Page IV D.1.20
		Rev. Date 6/70

is successful, the temperature of the isotope COMP block is changed to zero; if not, execution terminates. Using information from the LIMITS record, CRISP then calculates a factor used in determining the length of the SMAT block. Then subroutines are called to calculate the energy band structure and write the blocks CHI, XSEC, and SMAT on the LIDS.

3.8 Subroutine LGRPS

This very short routine reads the group structure record from MULTIGRP, computes the energy band widths, and stores the information in COMMON/GRUPS/.

3.9 Subroutine SIZE1

This subroutine is called when CRISP is used to set up the XSEC and SMAT blocks for the BL calculation. In this case, the module PASTE has the most severe core storage requirements of all the modules called by HOMBLE*. Therefore, the segmentation into energy bands is based on the amount of storage available in PASTE, rather than FIST. SIZE1 calculates the storage left after the following blocks are read into the DATA array:

```
'NAME'.NCASE.0 .0
'COMP'.NCASE.0 .0
'GEOM'.NCASE.0 .0
'QCEL'.NCASE.0 .0
'RESD'.NCASE.1 .0
```

This quantity LEFT is passed through common /SIZE/. (For details, see Section 2.8.)

3.10 Subroutine SIZE2

This subroutine is called when CRISP is executed by RAHAB. Since in this case the segmentation into energy bands is based on the amount of storage available in FIST, SIZE2 calculates the storage left after the fixed length blocks are read in. This quantity LEFT is passed through common /SIZE/. (See Section 2.8).

3.11 Subroutine SIZTM

This subroutine is used in calculating the size of the block TMAT. It returns two values to the calling program:

```
LTMATA - the constant part of TMAT
LTMATB - a multiplying factor
```

The length of the block for a given energy band is

$$LTMAT = LTMATA + LTMATB * G$$

* HOMBLE - Homogeneous Medium BL Calculation Equation

where G is the number of groups in the energy band.

3.12 Function LXSEC

This function calculates the size of the XSEC block for the energy band with lower limit IEBL and upper limit IEBH. See section 2.4 for the appropriate formula.

3.13 Function LSMAT

This function calculates the size of the SMAT block for the energy band with lower limit IEBL and upper limit IEBH. See section 2.6 for details.

3.14 Subroutine CALER

This subroutine calculates the energy band structure based on the storage available as indicated by LEFT. It then writes the groups structure record to the LIDS. The rules for selecting the energy bands are given in section 2.8.

3.15 Subroutine LMGXS

This subroutine reads the cross sections from MULTIGRP, loads them into the blocks XSEC and CHI and writes the blocks in the LIDS. The preparation of the XSEC and CHI blocks is described in section 2.9.

3.16 Subroutine LMGSM and DØDEBA

This subroutine loads scattering data from MULTIGRP into the SMAT blocks and writes them on the LIDS. The preparation of the SMAT blocks is described in Section 2.10. If detailed balance is to be used, the subroutine DØDEBA is called to compute and store the elements.

4. User Information

4.1 Input

CRISP cannot be used as a stand-alone module since it relies on the blocks NAME, COMP, GEOM, and ZONE written to the LIDS by ASP and LAGER and the block RESD written by RESIN. If these records have been created, CRISP requires only the record.

Name: 'INPUT'. 'CRISP'

List: DLAT, VLAT, TDSN, NCASE, XDSN, XVER, LPRNT, LPX, LEBC, NZNX, (IDØ(N), N=1, NZNX)

DLAT - Lattice name.

VLAT - Lattice version.

TDSN - Temporary data set name = 'TEMP'.

NCASE - Case number.

XDSN - Cross section data set name.

XVER - Cross section version.

LPRNT - If LPRNT=1, the maximum number of words used in the DATA array is printed.

LPX - Number of Legendre matrices to be processed.

LEBC - Flag indicating whether the storage available in PASTE or FIST will be used in the energy band calculation.

LEBC=1 CRISP is executed by HOMBLE, so PASTE storage is critical.

LEBC=2 CRISP is executed by RAHAB, so FIST storage is critical.

NZNX - Number of zones.

IDØ(N) - Flag for calculating size of TMAT

IDØ(N)=0, Calculate size of TMAT for zone N

IDØ(N)>0, Separate TMAT block for zone N is not given. Ignore calculation.

Usually this record is created by either the RAHAB or HOMBLE control module. These items are used to establish the names of the input and output records.

4.2 Output

If cross section and scattering data is available, the following blocks will be written on the LIDS:

'XSEC'. NCASE. NEB. 0

'SMAT'. NCASE. NEB. 0

'CHI' . NCASE. 0 . 0

Also, the group structure record is written:

Name: 'TEMP'.'LIDS'.DLAT.VLAT.'GRPS'.NCASE.O.O

List: NEG,NEBX,JFTG,JCHI,IDBF,CE(I),EBAR(I),DE(I),
I=1,NEG),E(NEG+1),(IEBH(N),N=1,NEBX)

NEG - Number of energy groups.

NEBX - Number of energy bands.

JFTG - Index to first thermal group.

JCHI - Highest group number at which CHI is given.

IDBF - Detailed balance flag.

E(I) - Energy break point (EV).

EBAR(I) - Characteristic energy (EV).

IEBH(N) - Highest group number in energy band N.

4.3 Core Size and Timing

The CRISP module requires 235K of core, 1 job data set block, and should be executed with a GØTIME=1 minute. Individual subprogram sizes and timings are given in Table I. The times were for uniform lattice with 7 isotopes.

4.4 Error Stops

All stops are handled by the JOSHUA operating system and are summarized in Table II.

JOSHUA DPSTM-500	Title Module CRISP	Vol. IV	Page D.1.24
		Rev.	Date 6/70

TABLE I

SIZE AND TIMING FOR CRISP SUBPROGRAMS

(Size in bytes, time in CPU seconds.)

<u>Subprogram</u>	<u>Size</u>	<u>Total Time</u>	<u>Average Time</u>	<u>Entries</u>
MAIN	3,794	2.32960	2.32960	1
LGRPS	724	.23296	.23296	1
SIZE1	292	-	-	-
SIZE2	1,134	.03328	.03328	1
SIZTM	5,430	.03328	.03328	1
CALEB	2,252	.04992	.04992	1
LXSEC	864	.34944	.00368	95
LMGXS	2,978	1.06496	1.06496	1
LSMAT	740	.03328	.00035	95
LMGSM	2,244	23.51232	23.51232	1
DODEBA	1,714	-	-	-
MANAGE	11,840*	1.68064**	.01474	114
IHCSEXP	432			
ICHSSQRT	329			
IHCSTATN2	472			
IHCSSCN	493			
/DATA/	60,000			
/SPECS/	128			
/TEMP/	43,216			
/SIZE/	24			
/SCRACH/	5,600			
/DATCAT/	2,440			
JOS	78,170***			
TOTALS	225,280	29.319		311

* Includes all routines of data manager.

** Includes time for JOSHUA Operating System and output routines.

*** Includes JOSHUA Operating System and buffers.

TABLE II

CRISP Error Stops

Message/Explanation

CRISP.MAIN,PØINTER MISSING

The input record is missing or the block NAME and CØMP cannot be read.

CRISP.MAIN,CØNTENTS RECØRD MISSING

The contents record for an isotope is not in the version of MULTIGRP specified in input record.

CRISP.MAIN,LIMITS XSECTION RECØRD MISSING

The cross-section limits record for an isotope is not in version of MULTIGRP specified in input record.

CRISP.MAIN, LIMITS TØTAL RECØRD MISSING

The limits-total record for an isotope is not in the version of MULTIGRP specified in input record.

CRISP.LGRPS, GRØUP STRUCTURE RECØRD MISSING

The groups record is not in the version of MULTIGRP specified in the input record.

CRISP.LGRPS, TØØ MANY GRØUPS

The number of groups exceeds 100.

CRISP.SIZE1, PØINTER MISSING

The GEØM,RESØ, or ZØNE record is missing.

CRISP.CALEB, INCORRECT BAND STRUCTURE

Programming error or JFTG is larger than the number of energy groups.

CRISP.LMGXS,CRØSS SECTION RECØRD MISSING

The cross sections for the isotope are not in the version of MULTIGRP specified in the input record.

JOSHUA DPSTM-500	Title Module CRISP	Vol.	Page
		IV	D.1.26
		Rev.	Date
			6/70

TABLE II (CONT.)

CRISP.LMGXS, ERRØR IN XSEC ØR CHI TABLES

Programming error.

CRISP.LMGSM, ERRØR IN SMAT TABLE

Programming error.

CRISP.SIZTM,ERRØR IN GEØMETRY TABLE

Either a sub-region or face number from the
ZØNE block cannot be found in GEØT.

CRISP.SIZTM,TØØ MANY SIDES.

Number of sides is greater than 500.

CRISP.SIZTM,GEØT BLØCK MISSING

The block GEØT is not in the LIDS.

CRISP.DØDEBA,PØINTER MISSING

SMAT block cannot be found.

CRISP.DØDEBA, ERRØR IN SMAT

Programming error.

Work Done By

C. E. Henderson
H. C. Honeck

Report Written By

H. C. Honeck

Contents

	<u>Page</u>
1. Theory	D.2.3
1.1 Introduction	D.2.3
1.2 Basic Method	D.2.3
1.3 Resonance Rate Adjustments	D.2.6
2. Program Logic	D.2.7
2.1 Interfaces	D.2.7
2.2 Numerical Approximations	D.2.7
2.3 Calculation Sequence	D.2.9
2.4 Storage Considerations	D.2.9
3. Coding Details	D.2.12
3.1 COMMON/DATA/	D.2.12
3.2 COMMON/SPECS/	D.2.12
3.3 COMMON/GRØUPS	D.2.13
3.4 COMMON/TEMP/	D.2.13
3.5 COMMON/SCRACH/	D.2.13
3.6 Main Program RESIN	D.2.13
3.7 Subroutine LRESC	D.2.13
3.8 Subroutine RESCAL	D.2.16
4. User Information	D.2.18
4.1 Input	D.2.18
4.2 Output	D.2.18
4.3 Core Size and Timing	D.2.18
4.4 Error Stops	D.2.18

JOSHUA DPSTM-500	Title Module RESIN	Vol. IV	Page D.2.2
		Rev.	Date 2/70

(Blank)

1. Theory

1.1 Introduction

Resonance reaction rates are stored on the RILIB data set by cell name, region name, isotope name, and resonance energy. For each resonance the data given are absorption rate, fission rate, and absorption shielding factor. Because these data are given by resonance energy and not by energy groups, they can be used with a variety of group structures. At some point in the RAHAB sequence of calculations the contributions from individual resonances must be converted to reaction rates for each energy group. If the conversion is done in each module where resonance rates are needed, the conversion will be repeated at least six times (twice in FIST, once in PASTE, and three times in RABED). The conversion procedure is rather complicated so that we have elected to perform it in a separate module, RESIN, which computes resonance reaction rates by cell, region, isotope, and energy group. These results (the RESD block) are written on the LIDS for use by later modules.

This procedure has the added benefit that it may be possible to make minor adjustments to the resonance rates without repeating the full resonance calculation (CREEP). For example, in a burnup calculation, many concentrations change slowly from time step to time step. Since shielding factors are included in the data, we can estimate the change in resonance rate caused by a small change in concentration using the NR1A formula. More sophisticated procedures can also be used if the parameters are included in the RILIB data set.

1.2 Basic Method

Resolved resonance rates on the RILIB data set are defined as the reaction rate (events in a given region, isotope, and resonance) owing to a unit slowing-down source (one neutron/sec.) in the cell. Unresolved resonance rates are similarly defined except that they are linearly connected (on a rate versus lethargy scale) distributions. The tabulated unresolved rates are signaled by negative energies and have units of events/unit lethargy. Typical data are illustrated in Figure 1.

We select some reference energy, E_0 , above the resonance region and use the lethargy variable $u = \ln(E_0/E)$. We further take the slowing-down source at E_0 to be unity ($q(0) = 1$). Define $R_a(u)$ as the total absorption rate/unit lethargy in the cell.^a The slowing-down source then obeys the equation

Figure 1. Schematic of the Resonance Rates in Group I and Various Regions of a Cell.

$$\frac{dq(u)}{du} = R_a(u) q(u) \quad (1)$$

which has the solution

$$q(u) = \exp \left[- \int_0^u R_a(u') du' \right] \quad (2)$$

The number of absorptions, r_a , which occur between lethargies u_1 and u_2 is

$$r_a(u_1, u_2) = \int_{u_1}^{u_2} du q(u) R_a(u) \quad (3)$$

Similarly the number of fissions is

$$r_f(u_1, u_2) = \int_{u_1}^{u_2} du q(u) R_f(u) \quad (4)$$

If u_1 and u_2 represent the lower and upper limits of an energy group, then the group reaction rates are defined by $r_a(u_1, u_2)/q(u_1)$ and $r_f(u_1, u_2)/q(u_1)$, that is, the reactions in the group per unit slowing-down source at the upper energy (lower lethargy) limit of the group.

Let m be an index which designates isotope/region pairs in the cell. The total reaction (absorption or fission) is a sum of the individual isotope/region contributions

$$R_a(u) = \sum_m R_a(u, m) \quad (5)$$

so that equations (2 - 4) become

$$q(u) = \exp \left[- \int_0^u du' \sum_m R_a(u', m) \right] \quad (6)$$

$$= \prod_m \exp \left[- \int_0^u du' R_a(u', m) \right] \quad (7)$$

$$r_a(u_1, u_2, m) = \int_{u_1}^{u_2} du q(u) R_a(u, m) \quad (8)$$

$$r_f(u_1, u_2, m) = \int_{u_1}^{u_2} du q(u) R_f(u, m) \quad (9)$$

The shielding factor, $S(u, m)$, is defined as the ratio of the actual absorption resonance integral to the infinite dilute absorption resonance integral. Thus, the infinite dilute rate is estimated from the expression

$$r_i(u_1, u_2, m) = \int_{u_1}^{u_2} du q(u) R_a(u, m)/S(u, m) \quad (10)$$

An average shielding factor for a group can then be defined as

$$S(u_1, u_2, m) = r_a(u_1, u_2, m)/r_i(u_1, u_2, m) \quad (11)$$

The quantities $r_a(u_1, u_2, m)/q(u_1)$, $r_f(u_1, u_2, m)/q(u_1)$, and $S(u_1, u_2, m)$ are computed by RESIN and output in the RESD block, where u_1 and u_2 correspond to energy group limits.

1.3 Resonance Rate Adjustments

If we assume that, over a small range of concentrations, the resonance shielding is given by the NRIA formula, we can estimate the resonance rate at a changed concentration. Let C denote concentration and subscripts 1 and 2 denote old and new values. The NRIA formula is

$$R_a/R_i = 1 / \sqrt{1 + \alpha C} = S \quad (12)$$

where α is a constant which depends weakly on C .

We can determine α from

$$\alpha = \frac{1}{C_1} \left[\left(\frac{1}{S_1} \right)^2 - 1 \right] \quad (13)$$

and the new shielding factor from

$$S_2 = 1 / \sqrt{1 + \alpha C_2} \quad (14)$$

The new reaction rates are given by

$$R_{a2} = R_{a1} (C_2 S_2 / C_1 S_1) \quad (15)$$

$$R_{f2} = R_{f1} (C_2 S_2 / C_1 S_1) \quad (16)$$

2. Program Logic

2.1 Interfaces

RESIN reads the following records,

('TEMP'. 'LIDS'. 'DLAT'. 'VLAT' → V)

'INPUT'. 'RESIN'

V. 'NAME'. NCASE. O. O

V. 'COMP'. NCASE. O. O

V. 'GEOM'. NCASE. O. O

'MULTIGRP'. VERSN. 'GRØUPS'

and the RILIB data set.

RESIN writes the record

V. 'RESD'. NCASE. O. O

2.2 Numerical Approximations

The integrals in equations (8 - 9) lead to error functions because of the assumed linearly connected representation of the unresolved rates. We can avoid evaluating these functions by using an approximate numerical method.

Consider an energy group whose limits are E_i and E_{i+1} . Take E_i as the reference energy so that the corresponding $u = 0$. The lethargy width of the group is $UMAX = \ln (E_i / E_{i+1})$.

Divide the group into equal lethargy width boxes numbered $N = 1, NX$. These are selected such that the box width, DU , is

$$\begin{aligned} DU &\approx 0.01 && \text{if } U_{MAX} \leq 5 \\ &= U_{MAX}/500 && \text{if } U_{MAX} > 5 \end{aligned} \quad (17)$$

The lower and upper lethargy limits of the N^{th} box are then

$$\begin{aligned} UN1 &= DU*(N-1) \\ UN2 &= UN1 + DU \end{aligned} \quad (18)$$

The total absorption rate, equation (5), can then be written

$$RA(N) = \sum_M RA(N,M) \quad (19)$$

If $R_a(M)$ is a resolved rate at energy E , then

$$RA(N,M) = R_a(M) \quad (20)$$

and N is the box in which E lies. For unresolved rates, we must first construct the lines connecting the unresolved points, and then integrate these linear segments over a box to obtain the total contribution to the box.

When all contributions have been added into $RA(N)$, the resonance escape probability, $P(N)$, can be calculated.

$$\begin{aligned} P(1) &= 1.0 \\ P(N) &= P(N-1)*[1 - RA(N)], \quad N = 2, NX \end{aligned} \quad (21)$$

In terms of the formulas in section 1,

$$P(u) = q(u)/q(u_1) \quad (22)$$

The desired group reaction rates are then

$$RAT(M) = \sum_{N=1}^{NX} P(N)*RA(N,M) \quad (23)$$

$$RFT(M) = \sum_{N=1}^{NX} P(N)*RF(N,M) \quad (24)$$

$$RIT(M) = \sum_{N=1}^{NX} P(N)*RA(N,M)/SF(N,M) \quad (25)$$

$$SFT(M) = RAT(M)/RIT(M) \quad (26)$$

2.3 Calculation Sequence

The over-all calculation sequence for RESIN is shown in Figure 2.

2.4 Storage Considerations

The order in which data is stored on the RILIB data set and the calculation required to obtain group rates demands that all of the data from RILIB for one cell be in core at one time. For large problems involving many isotope/region pairs and many resonances/isotope the data can be quite large. In some cases an isotope might have up to 200 resonances, but the average number will probably be 20 - 50. Therefore, a packed storage block in the DATA array is used to contain the input data from RILIB. The block, named RESC is defined in Figure 3. It is similar in structure to the other LIDS blocks and can be written to the LIDS. This might be desirable for burnup problems where RESIN is used repeatedly and the number of RILIB records is large.

Figure 2. Flow Diagram for Module RESIN.

RESC(K) KRPT(2,K)			
KCL+0	NCL	KNXTCL	CELL NO., POINTER TO NEXT CELL
1	IA	IB	GROUP LIMITS OF NON-ZERO DATA
2	NRIX		NO. OF RESONANCE ISOTOPES
KRI+0	NRI	KNXTRI	RES. ISO. NO., POINTER TO NEXT RES. ISO.
1	IRGX		NO. OF REGIONS
KRG+0	NRG	KNXTRG	REGION NO., POINTER TO NEXT REGION
1	IIS	KMT	ISO. NO. AND POINTER TO MAT'L. IN 'COMP' BLK.
2	IRSX		NO. OF RESONANCES
3	TDC		T(°C) FOR THE DATA
4	CON		CONCENTRATION (ATM/B-CM) FOR THIS DATA
5	ER(1,NRG)		} RESONANCE ENERGY CAPTURE RATE FISSION RATE SHIELDING FACTOR BY RESONANCE FOR THIS ISOTOPE IN THIS REGION
	RC(1,NRG)		
	RF(1,NRG)		
	SF(1,NRG)		

	ER(IRSX,NRG)		
	RC(IRSX,NRG)		
	RF(IRSX,NRG)		
	SF(IRSX,NRG)		

Figure 3. Schematic of the RESC Block

3. Coding Details

The RESIN module is divided into the following parts:

1. COMMON/DATA/
2. COMMON/SPECS/
3. COMMON/GRØUPS/
4. COMMON/TEMP/
5. COMMON/SCRACH/
6. Main program RESIN
7. Subroutine LRESC
8. Subroutine RESCAL
9. Subroutine MANAGE (See Manuscript B.1)

3.1 COMMON/DATA/

The LIDS blocks NAME, CØMP, GEØM, and RESD are packed in the array DATA defined by

```
COMMON/DATA/DATA(10000)      .
```

Refer to Manuscript B.1 for a description of this array and its usage.

The temporary block RESC is also stored in DATA. The symbol KRØRG and pointer array KRPT refer to the RESC block when used in subroutines LRESC and RESCAL. They refer to the RESD block when used in RESIN.

3.2 COMMON/SPECS/

The record 'INPUT'.'RESIN' contains the data:

TDSN	-	Temporary data set name = 'TEMP'
DLAT	-	Lattice name
VLAT	-	Lattice version
VERSN	-	MULTIGRP data set version name
NCASE	-	Case number
LPRNT	-	If LPRNT=1, maximum length of the DATA array used will be printed.

These data are stored in /SPECS/

3.3 COMMON/GRØUPS/

This area contains the group structure data from the MULTIGRP data set. The quantities used in RESIN are:

- NEG - number of energy groups
- E(I) - upper energy (ev) limit of group I.

3.4 COMMON/TEMP/

This area is used to transmit resonance rates for one group from the subroutine RESCAL to RESIN. The items in this area are:

- NFX - number of isotope/region pairs for this group
- NFXX - maximum value of NFX allowed = 100
- NRGT(N) - region number of the Nth isotope/region pair
- NIST(N) - isotope number of the Nth isotope/region pair
- RAT(N) - absorption rate of the Nth isotope/region pair
- RFT(N) - fission rate of the Nth isotope/region pair
- RIT(N) - infinite dilute absorption rate of the Nth isotope/region pair.

3.5 COMMON/SCRACH/

This is a scratch area used by LRESC to store input from RILIB and cross reference tables, and by RESCAL to store intermediate results.

3.6 Main Program RESIN

The flow diagram for RESIN is given in Figure 2.

3.7 Subroutine LRESC

This routine reads all required data from the RILIB and packs it into the RESC block. The flow diagram for LRESC is shown in Figure 4. Some additional comments are given below:

1. Cells, isotopes, and regions are referred to by name on the RILIB. Equivalent numbers are obtained by scanning the NAME, CØMP, and GEØM blocks.

JOSHUA DPSTM-500	Title Module RESIN	Vol. IV	Page D.2.14
		Rev.	Date 2/70

2. The CØNTENTS record on RILIB might contain names of isotopes/regions for which no data is actually given. While this is actually an error, LRESC ignores this and creates a new CØNTENTS record for the data actually given. This is done by subroutine TEMPP. (This feature will be eliminated at a later date.)
3. Resonance energies read from RILIB are automatically ordered by LRESC into descending energy order.

Figure 4. Flow Diagram for LRESC.

3.8 Subroutine RESCAL

This routine calculates resonance rates for one energy group and passes the results to RESIN through `COMMON/TEMP/`. The flow diagram for RESCAL is shown in Figure 5. Some comments are given below.

1. For each cell, two passes are used. The first pass accumulates the resonance rates in $RA(N)$, $RF(N)$, and $RI(N)$ according to equation (19) and then computes $P(N)$ from equation (21). In the second pass the arrays $RA(N)$, $RF(N)$, and $RI(N)$ contain the results for one isotope/region and the group rates are computed from equations (23 - 25).

Figure 5. Flow Diagram for RESCAL

JOSHUA DPSTM-500	Title Module RESIN	Vol. IV	Page D.2.18
		Rev.	Date 2/70

4. User Information

4.1 Input

RESIN cannot be used as a stand-alone module since it relies on the blocks NAME, COMP, and GEOM on the LIDS. Thus, the modules ASP and LAGER must have been previously executed. Appropriate data must also have been placed on the RILIB. If all of these records have been created, RESIN requires only the record:

Name: 'INPUT'. 'RESIN'

List: TDSN, DLAT, VLAT, VERSN, NCASE, LPRNT

TDSN - temporary data set name = 'TEMP'

DLAT - lattice name

VLAT - lattice version

VERSN - MULTIGRP data set version name

NCASE - case number

LPRNT - If LPRNT=1, the maximum length of the DATA array used will be printed.

Normally, this record is created by the RAHAB control module. These items are used to establish the names of the input and output records.

4.2 Output

If resonance data is found on the RILIB, the RESD block will be computed and written on the LIDS.

4.3 Core Size and Timing

The RESIN module requires 160K of core, 1 job data set block, and should be executed with a GØTIME = 1 minute. Individual sub-program sizes and timings are given in Table I. The timings were for a uniform lattice, 9 isotope/region pairs, and an average of 19 resonances/isotope.

4.4 Error Stops

All error stops are handled by the JOSHUA operating system and are summarized in Table II.

TABLE I

SIZE AND TIMING FOR RESIN SUBPROGRAMS
(Size in bytes, time in CPU seconds)

<u>SUBPROGRAM</u>	<u>SIZE</u>	<u>TOTAL TIME</u>	<u>AVERAGE TIME</u>	<u>ENTRIES</u>
MAIN	1,968	.38272	.38272	1
RESCAL	3,168	1.99680	.02377	84
LRESC	4,000	.54912	.54912	1
TEMPP	1,196	.18304	.18304	1
MANAGE	9,804*	.38272**	.000704	543
DATA	40,000			
TEMP	2,008			
SPECS	32			
GRØUPS	1,624			
DATCAT	2,440			
SCRACH	11,200			
IHCSSQRT	329			
IHCSLØG	442			
JØS***	77,437			
	<u>155,648</u>	<u>3.4944</u>		

* Includes all subroutines of data manager.

** Includes time for JOSHUA Operating System.

*** JOSHUA Operating System and buffers.

TABLE II

RESIN Error Stops

Message/Explanation

RESIN.MAIN, PØINTER MISSING

One or more of the blocks NAME, CØMP, GEØM, or GRPS, is not on the LIDS.

RESIN.MAIN, INPUT RECØRD MISSING

The record 'INPUT'.'RESIN' is missing.

RESIN.LRESC, PØINTER MISSING

Machine or program error.

RESIN.LRESC, INCØNSISTENT REGIØN NAMES

The region names used on RILIB do not agree with those in the NAME block for this lattice.

RESIN.LRESC, INCØNSISTENT ISØTØPE NAMES

The isotope names used on RILIB do not agree with those in the NAME block for this lattice.

RESIN.LRESC, DATA ARRAY ØVERFLØW

The data from the RILIB cannot be contained in the DATA array.

RESIN.RESCAL, PØINTER MISSING

Machine or program error.

RESIN.RESCAL, RESØNANCE TABLES ØVERFLØW

There are too many isotope/region pairs. Increase the size of the tables in CØMMØN/TEMP/.

Work Done By

D. A. Sharp

Report Written By

D. A. Sharp

Contents

	<u>Page</u>
1. Theory	D.3.3
1.1 Introduction	D.3.3
1.2 Computation of Resolved Resonance Integrals	D.3.3
1.3 Reaction Rates	D.3.8
1.4 Computation of Unresolved Resonance Integrals	D.3.9
1.5 Numerical Approximations	D.3.11
1.5.1 Resolved Resonances	D.3.11
1.5.2 Collision Probability Computation	D.3.12
1.5.3 Doppler Broadened Cross Sections	D.3.14
1.5.4 Unresolved Resonances	D.3.15
2. User Information	D.3.17
2.1 Interfaces with RAHAB	D.3.17
2.2 Computational Sequence	D.3.19
2.3 Subroutine Descriptions	D.3.19
2.4 Coding Details	D.3.23
2.4.1 COMMON Areas	D.3.23
2.4.2 Storage Considerations	D.3.25
2.4.3 Error Stops	D.3.25

JOSHUA DPSTM-500	Title Module CREEP	Vol. IV	Page D.3.2
		Rev.	Date 6/70

(Blank)

1. Theory

1.1 Introduction

The CREEP module is used to compute resolved and unresolved resonance integrals for all isotopes in a lattice with resonance parameters. Computation is done on a cell-wise basis, each cell in the lattice being treated in sequence. Output from the module is in the form of unresolved and resolved reaction rates for each resonance absorber. These reaction rates are normalized to one neutron per second slowing down in the cell. Both absorption and fission rates are computed and stored in the RILIB data set for each individual resonance, and for each region in the cell containing the absorber.

An assumption basic to the treatment is that resonance absorption can be computed for each cell individually in the lattice, without taking into account interactions with neighboring cells. In addition to saving computation time, this permits the reuse of previously computed cell results in other lattice calculations.

1.2 Computation of Resolved Resonance Integrals

The effective resonance integral, I_i , is defined as being that microscopic cross section which, when multiplied by the asymptotic flux in the i^{th} absorber region of a cell, and by the number of absorber atoms in that region, yields the neutron absorption rate due to one resonance in the absorber, i.e.,

$$N_i V_i \phi_i^{\text{AS}} I_i = \int_{\text{resonance}} dE \int_{V_i} d\vec{r}_i \phi(\vec{r}_i, E) \Sigma_a(\vec{r}_i, E) \quad (1)$$

If $\Sigma_a \neq \Sigma_a(\vec{r}_i)$ in the i^{th} region, the effective resonance integral is

$$I_i = \frac{1}{\phi_i^{\text{AS}} N_i} \int_{\text{resonance}} dE \Sigma_a(E) \left(\frac{1}{V_i} \int_{V_i} d\vec{r}_i \phi(r_i, E) \right) \quad (2)$$

where the bracketed term is the volume-averaged flux in the i^{th} region, i.e.,

$$\bar{\phi}_i = \frac{1}{V_i} \int_{V_i} d\vec{r}_i \phi(r_i, E) \quad (3)$$

If we consider the cell to be made up of J absorber and J' moderator regions, then equation (3) may be expanded to

$$\begin{aligned} \phi_1(E) = \frac{1}{V_1} \int_{V_1} d\vec{r}_1 \left\{ \sum_{j=1}^J \int_{V_j} d\vec{r}_j T(\vec{r}_1, \vec{r}_j, E) S_j(\vec{r}_j, E) \right. \\ \left. + \sum_{j'=1}^{J'} \int_{V_{j'}} d\vec{r}_{j'} T(\vec{r}_1, \vec{r}_{j'}, E) S_{j'}(\vec{r}_{j'}, E) \right\} \end{aligned} \quad (4)$$

In equation (4), $T(\vec{r}_1, \vec{r}_j, E)$ is the probability that neutrons emerging from collisions in $d\vec{r}_j$ with energy E will subsequently travel to $\vec{r}_1$ in the jth absorber region without suffering further collisions.

The source of scattered neutrons with energy E, $S_j(\vec{r}_j, E)$, may, for elastic scattering events, be expressed as

$$S_j(\vec{r}_j, E) = \int_{E' > E} \phi(\vec{r}_j, E') \Sigma(\vec{r}_j, E' \rightarrow E) dE' \quad (5)$$

where $\Sigma(\vec{r}_j, E' \rightarrow E)$ expresses the probability of energy transfer from E' to E for an elastic scattering event.

Sources in the moderator regions are treated using the "narrow resonance" approximation, while the "infinite absorber" approximation is applied to those in the absorber regions.

These approximations, when applied to equation (5) yield

$$S_{j'} = \frac{\Sigma_s(E)}{E} \phi_{j'} \quad (6)$$

for the moderator regions, and

$$S_j = \phi_j \Sigma_{s,j}(E) \quad (7)$$

for the fuel region, where the fluxes ϕ_j and $\phi_{j'}$ are assumed to vary spatially, from region to region.

Substitution of equations (6) and (7) into (4) yields

$$\begin{aligned} \bar{\phi}_i(E) = & \sum_{j=1}^J \frac{S_j(E)}{V_1} \int_{V_1} d\vec{r}_1 \int_{V_j} d\vec{r}_j T(\vec{r}_1, \vec{r}_j, E) \\ & + \sum_{j'=1}^{J'} \frac{\Sigma_s(E)}{E} \frac{\phi_{j'}}{V_1} \int_{V_1} d\vec{r}_1 \int_{V_{j'}} d\vec{r}_{j'} T(\vec{r}_1, \vec{r}_{j'}, E) \end{aligned} \quad (8)$$

If the probability of collision in region 1 of a neutron coming from region j is defined as

$$P_{1j} = \frac{\Sigma_{T1}}{V_j} \int_{V_1} d\vec{r}_1 \int_{V_j} d\vec{r}_j T(\vec{r}_1, \vec{r}_j, E) \quad (9)$$

then equation (8) becomes, with the substitution,

$$\bar{\phi}_i(E) = \frac{1}{\Sigma_{T1}} \left\{ \sum_{j=1}^J S_j(E) \frac{V_j}{V_1} P_{1j} + \sum_{j'=1}^{J'} \frac{\Sigma_s}{E} \phi_{j'} \frac{V_{j'}}{V_1} P_{1j'} \right\} \quad (10)$$

and the effective resonance integral is

$$\begin{aligned} I_1 = & \frac{1}{\phi_1^{AS} N_1} \int_{\text{resonance}} dE \frac{\Sigma_{A1}}{\Sigma_{T1}} \left\{ \sum_{j=1}^J S_j \frac{V_j}{V_1} P_{1j} \right. \\ & \left. + \sum_{j'=1}^{J'} \frac{\Sigma_s}{E} \phi_{j'} \frac{V_{j'}}{V_1} P_{1j'} \right\} \end{aligned} \quad (11)$$

In CREEP calculations the fuel source terms, S_j , are represented by a single term $\bar{S}_0$, computed at the cell-averaged number density of the absorber isotope, i.e.,

$$\bar{S}_0 = \phi_1^{AS} \bar{\Sigma}_{sf}, \quad (12)$$

where

$$\bar{\Sigma}_{sf} = \sigma_{sf} \frac{\sum_{j=1}^J N_j V_j}{\sum_{j=1}^J V_j} \quad (13)$$

In order to account for this approximation, correction factors f_j are computed for each absorber region, so that $f_j \bar{S}_0 \approx S_j$. These are of the form

$$f_j = 1 - \frac{(\bar{\Sigma}_{sf} - \Sigma_{sfj})}{\bar{\Sigma}_{sm} + \bar{\Sigma}_{sf}} \quad (14)$$

where $\bar{\Sigma}_{sm}$ is the cell-averaged scattering cross section of moderators "admixed" in the absorber regions.

With this approximation, and with the assumption of a spatially flat asymptotic flux, so that $\phi_j = \phi_j$, equation (11) becomes

$$I_i = \int_{\text{resonance}} dE \frac{\Sigma_{A_i}(E)}{\Sigma_{T_i}(E)} \left\{ P_{oi} \bar{S}_0(E) + \frac{P_{li}}{E} \right\} \quad (15)$$

with

$$P_{oi} \equiv \sum_{j=1}^J \frac{V_j}{N_i V_i} f_j P_{ij} \quad (16)$$

and

$$P_{li} \equiv \sum_{j'=1}^{J'} \frac{V_{j'}}{N_i V_i} \Sigma_{sj'} P_{ij'} \quad (17)$$

In CREEP, equation (15) is evaluated for each resonance, using techniques developed by Nordheim¹. The fuel scattering source, $\bar{S}_0$, is developed as a sum of terms, including neutron scattering contributions both from the resonance absorber and from two admixed moderator materials. The expression for $\bar{S}_0$, in lethargy notation, is

1. L. W. Nordheim, A Program of Research and Calculations of Resonance Absorption. GA-2527, August 25, 1961.

$$\begin{aligned} \bar{S}_0(u) = & \frac{1}{\alpha_f} \int_{u-\Delta_f}^u \phi(u') \sigma_{s_f}(u') du' + \frac{1}{\alpha_{m_1}} \int_{u-\Delta_{m_1}}^u \phi(u') \sigma_{s_{m_1}}(u') du' \\ & + \frac{1}{\alpha_{m_2}} \int_{u-\Delta_{m_2}}^u \phi(u') \sigma_{s_{m_2}}(u') du' \end{aligned} \quad (18)$$

with the following definitions of subscripts and terms:

- $f \rightarrow$ absorber atoms
- $m_1 \rightarrow$ admixed moderator with lowest mass number
- $m_2 \rightarrow$ most abundant admixed moderator
- $\alpha_1 = (A_1 - 1)^2 / (A_1 + 1)^2$
- $\Delta_1 = \ln(1/1 - \alpha_1)$

Details of the computational procedures followed in CREEP to evaluate equations (15) and (18) are given in section 1.4.2.

JOSHUA DPSTM-500	Title Module CREEP	Vol.	Page
		IV	D.3.8
		Rev.	Date
			6/70

1.3 Reaction Rates

The resonance integrals I_i are cast into the form of reaction rates per unit source strength before being written on the RILIB. The total reaction rate, R , in the i^{th} absorber region due to a particular resonance is, from equation (1),

$$R \equiv \phi N_i V_i I_i \text{ reactions/second} \quad (19)$$

with a spatially independent flux, ϕ .

The flux distribution due to a unit slowing down source Q_0 in the cell is defined by

$$\phi(u) \bar{\xi} \Sigma_s V \equiv Q_0 \equiv 1 \quad (20)$$

$$\text{where } \bar{\xi} \Sigma_s V \equiv \sum_{i \text{ regions}} (\bar{\xi}_i \Sigma_{s_i} V_i)$$

and $\bar{\xi}_i$ = average change in lethargy per collision in the material in the i^{th} region

$$= \frac{1}{\Sigma_{s_i}} \sum_{j \text{ isotopes}} \xi_j \Sigma_{s_j}$$

$$\text{Thus, } R = \frac{N_i V_i I_i}{\bar{\xi} \Sigma_s V} \quad (21)$$

Both absorption and fission rates are computed and stored in RILIB.

1.4 Computation of Unresolved Resonance Integrals

Unresolved resonance absorption data are stored on the RILIB data set as region-wise reaction rates per unit lethargy for each isotope. Computations are done for neutrons with zero angular momenta. The procedure followed in the calculation is essentially that of Nordheim², wherein the total unresolved resonance integral for an isotope in a cell is represented by

$$I_u = \frac{\sigma_p \Gamma_r}{D} \int_{\Delta E} \bar{J}(E) \frac{dE}{E} \quad (22)$$

where D is the level spacing, σ_p the potential cross section per absorber atom, Γ_r the absorption width, and $\bar{J}(E)$ is an average of the J^r function over the Porter-Thomas distribution of reduced neutron widths. The integral (22) spans the total unresolved energy region.

In CREEP, this energy region is subdivided, on a lethargy basis, into N units of width $\Delta u = 0.25$. The integrand of equation (22) is then computed at the points $u_j, j=0, \dots, N$, yielding $N+1$ "ordinates" of the form

$$Z_j = \frac{\sigma_p \Gamma_r}{D} \bar{J}_j(u_j) \quad , \quad j = 0, \dots, N \quad (23)$$

The Z_j are then split into components in order to estimate the spatial dependence of the resonance integral. This is done by assigning each absorbing region a weight equal to the ratio of its collision probability relative to the cell average, i.e.,

for the i^{th} region:

$$w_i = \frac{K_i}{\bar{K}} \quad (24)$$

where

$$K_i = \frac{N_i}{\sum_{T1}} P_{11}$$

and

$$\bar{K} = \frac{\sum_{\substack{i \text{ absorber} \\ \text{regions}}} N_i V_i K_i}{\sum_i N_i V_i}$$

2. L. W. Nordheim, A Program of Research and Calculations of Resonance Absorption. GA-2527, August 25, 1961.

JOSHUA DPSTM-500	Title Module CREEP	Vol. IV	Page D.3.10
		Rev.	Date 6/70

The quantities $Z_{ij} = w_i Z_j$ are then converted to reaction rates by the treatment in section 1.3 and are stored on the RILIB data set by isotope, region, and energy. Their presence in the data set is signaled by the negative energies, $-E_j$, $j = 0, \dots, N$. Numerical evaluation procedures for the J_j are presented in section 1.5.4.

1.5 Numerical Approximations

1.5.1 Resolved Resonances

Integrals of the form of equation (18) must be evaluated for each resolved resonance calculation. In the calculation, a Simpson's rule technique is used that permits a step-by-step evaluation of the integrands of (18). The basic procedure is as follows:

- a. For each resonance, suitable upper and lower lethargy limits are defined, in order to span the resonance peak.
- b. The lethargy interval between these limits is subdivided, using a mesh spacing related to the Doppler width of the resonance at 273°K, as follows:

$$\epsilon = \frac{\Gamma_{D_0}}{rE_0} \quad (25)$$

where

Γ_{D_0} = Doppler width at 273°K

E_0 = resonance peak energy

r = a reduction factor equal to

$$\left(5 + \frac{\Gamma_{D_0}}{4} \right)$$

where Γ is the total width

K = an input scale factor, which can be used to arbitrarily increase the mesh width.

- c. The total number of mesh intervals, N_1 , over which a particular integral in equation (18) is to be evaluated is defined by

$$N_1 \epsilon \equiv \Delta_1 = \ln \left(\frac{1}{1 - \alpha_1} \right) \quad (26)$$

Thus, the slowing-down source at lethargy u_k under the resonance is given by

$$\begin{aligned} \bar{S}_{o_k} = & \frac{1}{\alpha_o} \int_{u_k - N_o \epsilon}^{u_k} \phi(u') \sigma_{s_o}(u') du' + \frac{1}{\alpha_{m_1}} \int_{u_k - N_1 \epsilon}^{u_k} \phi \sigma_{s_{m_1}} du' \\ & + \frac{1}{\alpha_{m_2}} \int_{u_k - N_2 \epsilon}^{u_k} \phi \sigma_{s_{m_2}} du' \end{aligned} \quad (27)$$

- d. The $\bar{S}_{o_k}$ are evaluated sequentially, using Simpson's rule, for all the mesh points under the resonance. The procedure starts at the lowest lethargy limit ($u=0$); $\bar{S}_o$ is evaluated assuming an asymptotic flux behavior ϕ_o at lower lethargies. Subsequently, a recursive relationship is formed to compute the $\bar{S}_{o_l}$, $l = 1, \dots, L$.
- e. Evaluation of equation (15) for the resonance integral is done using the trapezoidal rule, over the same mesh intervals. Accumulation of the integrand terms of (15) takes place concurrently with the $\bar{S}_{o_l}$ evaluations.
- f. Numerical approximations are made in order to estimate the absorption contribution made by the wings of the resonance. Details of this procedure are described by Nordheim³ and Kuncir⁴.

1.5.2 Collision Probability Computation

The collision probabilities for each absorber region, P_{o_i} and P_{1i} , $i = 1, \dots, J$, must be computed during the course of the integration procedure just described, for all mesh points under the resonance. This is done using approximate expressions for transmission probabilities between annular regions in a cell, under the assumption that the angular distribution of the neutrons crossing region boundaries is cosine-shaped. Since explicit

3. L. W. Nordheim, A Program of Research and Calculations of Resonance Absorption. GA-2527, August 25, 1961.
4. G. F. Kuncir, A Program for the Calculation of Resonance Integrals, GA-2525, August 26, 1961.

computation of these quantities for all mesh points is a time-consuming procedure, an optional interpolation technique is included in CREEP for their evaluation. Procedures used in this technique are as follows:

- a. For a fixed cell geometry and moderator composition, the collision probabilities are functions of the total cross sections in the regions containing the resonance absorber. In CREEP, before the resolved resonance calculation is done, a range, $\Delta\Sigma_T$, of absorber cross section is arbitrarily defined for the innermost (radially) absorber region. This range is picked so as to bound, approximately, the expected total cross section variation over the resonance peak.
- b. Similar $\Delta\Sigma_T$ ranges are defined for the other cell regions containing the absorber, based upon the region-to-region change in absorber number density.
- c. These ranges are broken into ten equal intervals; within each such interval, nine subintervals are defined. The P_{0i} and P_{1i} are then computed at each of the ten equally spaced cross section points in the interval.
- d. These values are then fitted, by least squares, to equations cubic in the absorber cross section. Equations are computed for each cross section interval, i.e., for a cell with three absorber regions, CREEP will compute a total of $10 \times 2 \times 3 = 60$ sets of coefficients.
- e. At each computation of Doppler-broadened cross sections in the resolved calculation (each lethargy mesh point), the total cross section in each absorber region is tested to see if it lies within the $\Delta\Sigma_T$ interval for the region. If so, the proper cubic equations are evaluated for the P_{0i} and P_{1i} . If not, collision probability calculations are done explicitly.

1.5.3 Doppler Broadened Cross Sections

Doppler broadened absorption and scattering cross sections are computed from

$$\sigma_a(E) = \frac{\Gamma_r}{\Gamma} \sqrt{\frac{E_0}{E}} \psi(\xi, X) \quad (28)$$

and

$$\sigma_s(E) = \frac{\Gamma_n}{\Gamma} \sigma_o \psi(\xi, X) + \left(\sigma_p \sigma_o \frac{\Gamma_r}{\Gamma} g \right)^{1/2} \chi(\xi, X) + \sigma_p$$

where

Γ = total width of the resonance
 Γ_n = scattering width

Γ_r = absorption width

E_0 = peak resonance energy

σ_o = peak resonance cross section

σ_p = potential scattering cross section

g = spin factor

ξ = Γ/Γ_D , where Γ_D is the "Doppler width" of the resonance

X = $\frac{2}{\Gamma} (E - E_0)$

ψ = $\frac{\xi}{\sqrt{4\pi}} \int_{-\infty}^{\infty} \frac{dy}{1+y^2} e^{-\xi^2(X-y)^2/4}$

χ = $\xi/\sqrt{4\pi} \int_{-\infty}^{\infty} \frac{2ydy}{1+y^2} e^{-\xi^2(X-y)^2/4}$

Procedures used to evaluate the ψ and χ functions vary with the ranges of the arguments as follows:

e.g. $\left| \frac{\xi X}{2} \right| \leq 4.0$

For this argument range, series expansions of the

incomplete gamma function are used to evaluate ψ and χ . Either asymptotic or convergent expansions may be used, depending upon the relative magnitudes of ξ and X .

b. $\left| \frac{\xi X}{2} \right| > 4.0$

For this range, the ψ and χ functions are expressed in terms of the real and imaginary parts of the complex error function. A polynomial approximation is used to evaluate the error function for various values of ξ and X .

1.5.4 Unresolved Resonances

The $\bar{J}$ function in equation (23) is evaluated at each mesh point in the unresolved region using a Simpson's rule integration procedure. The integral evaluated is

$$\bar{J}(E) = \frac{1}{\sqrt{2\pi G}} \int_B^{\infty} \frac{d\xi}{\sqrt{\xi-B}} J(\xi, B) e^{-(\xi-B)/2G} \quad (30)$$

where

$$G = \langle \Gamma_n^o \rangle \sqrt{\frac{A_o}{4kT}} \quad , \quad B = \Gamma_r \sqrt{\frac{A_o}{4EkT}}$$

$$H = \frac{\sigma_p E}{2.6(10^6)g} \quad , \quad B = \frac{H\xi}{\xi-B}$$

and

$$\langle \Gamma_n^o \rangle = \text{average neutron width}$$

$$\Gamma_r = \text{radiation width}$$

$$g = \text{spin factor}$$

$$\sigma_p = \text{potential cross section}$$

$$A_o = \text{mass of the absorber}$$

$$\xi = \frac{\Gamma}{\Gamma_D}$$

Equation (30) is evaluated using tabular values of the J function contained in the CREEP module. The function is tabulated as $J(\xi, K(\xi))$, where

$$K(\xi) = \frac{\ln \frac{H\xi}{\xi-B} + 5 \ln 10}{\ln 2} \quad (31)$$

Detailed interpolation procedures used in the J tables are presented by Kuncir⁵.

-
5. G. F. Kuncir, A Program for the Calculation Resonance Integrals, GA-2525, August 26, 1961.

2. User Information

2.1 Interfaces with RAHAB

The following JOSHUA records are read by the CREEP module:

1. 'INPUT'. 'RESCALC'

List: ALATT = Lattice name (R*8)
 AVERSN = Lattice version (R*8)
 TPFIT = Collision probability interpolation option (Yes or No).
 NCASE = Lattice case number.
 TDSN = Lattice data set name.
 FVER - Version of the NUCPARAM library.
 ANEWD - Lethargy mesh multiplication factor (K in Section 1.5.1).

Remarks: This record is user-supplied. It contains definitions of computational options, along with data set names necessary to access the resonance parameter library and the NAME and CARD arrays.

2. TDSN. 'LIDS'. ALATT. AVERSN. 'CARD'. NCASE. 0.0

3. TDSN. 'LIDS'. ALATT. AVERSN. 'NAME'. NCASE. 0.0

Remarks: These records are read to supply lattice material and geometry descriptions generated by the ASP and LAGER modules.

4. 'RILIB'. DCEL. VCEL. 'CØNTENTS'

5. 'RILIB'. DCEL. VCEL. 'TØTALS'

Remarks: These records are read to check the presence of results that have been previously computed for the cell whose name and version are denoted by DCEL.VCEL. If these records are found in RILIB, all calculations will be bypassed.

JOSHUA DPSTM-500	Title Module CREEP	Vol. IV	Page D.3.18
		Rev.	Date 6/70

6. 'NUCPARAM'.FVER.'SIGMAP'

Remarks: This record is read to obtain potential cross sections for isotopes in the cell.

7. 'NUCPARAM'.FVER.DISØ(I). 'UNRESØLV'.KZAS(I)

Remarks: This record contains unresolved resonance parameters for the Ith isotope in the cell.

8. 'NUCPARAM'.FVER.DISØ(I). 'RESØLVED'.KZAS(I)

Remarks: This record contains resolved resonance parameters for the Ith isotope in the cell. Records 7 and 8 also serve as a test of the presence of resonance absorption in the Ith isotope.

The following JOSHUA records are written by the CREEP module.

1. 'RILIB'.DCEL.VCEL.'CØNTENTS'

List: Number of entries in this record; isotope name, region name, region temperature and isotope concentration for each entry.

2. 'RILIB'.DCEL.VCEL.DISØ(I).DREGØ(N). 'RESULTS'

List: For the Ith isotope and the Nth region: number of resonance energies, resonance energies, resonance absorption rates, resonance fission rates, shielding factors.

Remarks: Records 1 and 2 are subsequently read by the RESIN module.

3. 'RILIB'.DCEL.VCEL.'TØTALS'

List: For each isotope having resonance absorption: isotope identification number, effective resonance integral, infinite resonance integral, shielding factor.

2.2 Computational Sequence

The sequence in which operations are performed in CREEP is illustrated in Figure 1.

2.3 Subroutine Descriptions

The CREEP module consists of a main program and twenty-seven subroutines. Brief descriptions of the functions of these routines follow.

MAIN Program

Lattice data is read here, from the CARD and NAME arrays. Each cell in the lattice is treated sequentially, first with a check on the presence of previously computed results in RILIB. If no results exist, geometry and material data is extracted from the CARD array and resonance computations proceed with control passing to subroutine ENTER.

Subroutine ENTER

This routine first tests isotope and material lists to determine if names are consistent with library data. Then, it loops over all isotopes, testing for the presence of resolved and/or unresolved resonance parameters. All JOSHUA output records are written by this routine.

Subroutine RCD

This routine updates pointers to the CARD array and returns values of cell parameters to the MAIN program.

Subroutine FIND

This routine compares regionwise material and isotope name lists for each cell with the material list and with the NUCPARAM isotope name list respectively. Any inconsistency generates an error stop.

Subroutine NIT

This routine is the control program for the resolved resonance calculation. Here, the resonance counter is initialized and the collision probability input option is tested.

Figure 1. Flow Diagram for Module CREEP

Subroutine GRID

This routine computes the mesh width to be used in the numerical integration over each resolved resonance, along with the energy limits of the integration.

Subroutine RESOL

This routine does the numerical integration to compute the resolved resonance integral for each region; wing correction calculations are included here.

Subroutine CRØSS

This routine computes Doppler broadened cross sections at each mesh point under the resolved resonance, for

$$\left| \frac{gX}{2} \right| \leq 4.0.$$

Subroutine QIKPSI

This routine computes the Doppler broadened cross sections for

$$\left| \frac{gX}{2} \right| > 4.0.$$

Subroutine GAMINC

This routine evaluates the incomplete gamma function for use in CRØSS.

FUNCTION Subprograms SAMPS, SEMPS, SØMPS

These routines perform Simpson's rule integrations over the slowing-down sources in the resonance absorber, the admixed moderator with lowest atomic weight, and the most abundant admixed moderator, respectively.

Subroutine ZUTIN

In this routine, cell-averaged concentrations and temperatures are defined for each isotope. Also, the admixed moderators are selected and the potential cross section per absorber atom is computed for each one.

Subroutine PPTBL

This routine is the control program for the collision probability computation. Here, input parameters for PZCALC and ESCAPE are generated; the regionwise collision probabilities are stored in a scratch COMMON area, and the unresolved resonance splits are computed.

JOSHUA DPSTM-500	Title Module CREEP	Vol. IV	Page D.3.22
		Rev.	Date 6/70

Subroutine PZCALC

This routine computes the collision probabilities P_{0i} and P_{1i} for each of the i absorber regions, and for each mesh point under the resolved resonance.

Subroutine ESCAPE

This routine computes region-to-region neutron transmission probabilities for use in PZCALC.

Subroutine QIKPRB

This is the control routine for the collision probability interpolation procedure. For each isotope, the cross section range and the intervals within the range are defined. Coefficients of the polynomials in the cross section are stored in COMMON for later use in COLLID.

Subroutine LSFIT

This routine is used to fit, by least squares, the collision probabilities P_{0i} and P_{1i} for each absorber region and for each cross section interval.

Subroutine COLLID

This routine optionally evaluates cubic equations in the absorber cross section for the collision probabilities at each resolved resonance mesh point.

Subroutine TUZ

This is the control program for the unresolved resonance computation. Here, the number of mesh points in the unresolved region is computed, and the quantities Z_j (equation 23) are evaluated.

Subroutine SIMPS

This routine computes the J function at each lethargy mesh point in the unresolved region, using a Simpson's rule integration procedure.

Function BARJ

This routine is used to evaluate the integrand of the J function evaluated in SIMPS.

Function FJ

This routine is used to interpolate in tables for $J(\xi, K(\xi))$.

Function Subprograms TXI, TK, TJ

These routines are used in conjunction with FJ, to interpolate for the J function. Routines TXI and TK return the tabular values of ξ and $K(\xi)$ respectively. Routine TJ returns the value of $J(\xi, K)$. Values of J are extracted from one of two COMMON blocks, depending upon the size of K.

Subroutine OUTPUT

This routine is used to compute the reaction rate for each region, and at each energy point. Unresolved rates are labeled by negative energies. Rates are accumulated in the JJOUT common area during the course of the calculation.

2.4 Coding Details

2.4.1 COMMON Areas

The following labeled COMMON blocks are contained in the CREEP module.

COMMON

Function

/DATAIN/

Used to transfer cell characteristics between MAIN and ENTER. Contains region, material, and isotope names, along with all dimensions, volumes and regionwise isotope concentrations.

/CARD/

Contains the CARD array.

/DDATA/

Contains the NAME array.

/BIGCOM/

Used to transfer resonance parameters for each isotope between portions of resolved resonance integral calculation; serves as a scratch data set for storage of the collision probabilities.

/JJOUT/

Used to store resonance energies, absorption and fission rates, and shielding factors for each isotope.

COMMON

Function

/UNRES/

Used to transfer resonance parameters for each isotope between portions of the unresolved resonance calculation.

/STORE/

The storage location of the coefficients of the collision probability polynomials.

/PASS/

Used to pass cross section parameters between RESOL, CROSS, and QIKPSI

/HOLD/

Used to pass miscellaneous data between subroutines involved in the resolved resonance calculation. Included are admixed moderator cross sections and mass numbers, lethargy mesh widths, resonance energy bounds, and partial and total resonance widths.

/A/ & /B/

These contain tabular values of the J function.

JOSHUA DPSTM-500	Title Module CREEP	Vol. IV	Page D.3.25
		Rev.	Date 6/70

2.4.2 Storage Considerations

The following bound on problem size are set by storage limitations in CREEP.

Maximum number of isotopes per cell	=	30
Maximum number of regions per cell	=	20
Maximum number of materials per cell	=	15
Maximum number of resonances per isotope.	=	200
Maximum number of regions per cell containing resonance absorbers.	=	5

2.4.3 Error Stops

Several input conditions will lead to termination of CREEP execution. These are listed below. Messages occur as printed output.

Message/Explanation

1. ERROR IN INPUT.RESCALC RECORD; EXECUTION IS TERMINATED/The INPUT.RESCALC record is missing.
2. ERROR IN LATTICE.???.CARD.???.?RECORD/The record TDSN.ALATT.AVERSN.'CARD'.NCASE.O.O is missing. Values of ALATT,AVERSN, and NCASE are printed with this message.
3. ERROR IN LATTICE.???.NAME.???.?RECORD/The record TDSN.ALATT.AVERSN.'NAME'.NCASE.O.O is missing. Values of ALATT, AVERSN, and NCASE are printed with this message.
4. ERROR IN FINEGRP.OPENHAMR.'SIGMAP'RECORD - INPUT/The record 'NUCPARAM'.FVER.'SIGMAP' is missing.
5. ERROR IN MATERIAL NAMES/The material name assigned to a particular region cannot be matched with any name in the materials list. The materials list and the list of materials by region are printed with this message.
6. ERROR IN ISOTOPE NAMES/An isotope name used in the problem cannot be matched with any entry in the NUCPARAM library. The isotope list in the NUCPARAM library is printed with this message, along with the list of isotope names used in the problem.

JOSHUA

DPSTM-500

Title Part E - Integral Transport Modules

Vol. IV	Page E.O.1
Rev.	Date 6/70

Contents

- E.1 Transport Matrix Processor - Module TRAMP
- E.2 Fine-Structure Calculation - Module FIST

JOSHUA
DPSTM-500

Title Part E - Integral Transport Modules

Vol.

IV

Page

E.O.2

Rev.

Date

6/70

(Blank)

Work Done By

C. E. Henderson
H. C. Honeck

Report Written By

C. E. Henderson
H. C. Honeck

Contents

	<u>Page</u>
1. Theory	E.1.3
1.1 Introduction	E.1.3
1.2 Transport Equation	E.1.3
1.3 Direct Integration Method	E.1.6
1.4 Cosine Currents Method	E.1.12
2. Program Logic	E.1.15
2.1 Interfaces	E.1.15
2.2 Over-all Structure	E.1.15
2.3 Zone Constants (ZNCØN,MTXS)	E.1.18
2.4 Symmetry Considerations	E.1.18
2.5 Numerical Integration	E.1.23
2.6 Tracking Routines (TRACK,TRACK1,TRACK2)	E.1.25
2.7 Transport Matrix Generators (TMG1,TMG2)	E.1.26
2.8 Evaluation of the P(α ,y) Functions (TXX)	E.1.27
2.9 Final Corrections to the Matrices (FIXTM)	E.1.28
3. Coding Details	E.1.28
3.1 CØMMØN/DATA/	E.1.29
3.2 CØMMØN/SPECS/	E.1.29
3.3 CØMMØN/REFPNT/	E.1.29
3.4 CØMMØN/TRAK/	E.1.30
3.5 CØMMØN/TEMP/	E.1.30
3.6 CØMMØN/SCRACH/	E.1.31
4. User Information	E.1.32
4.1 Input	E.1.32
4.2 Output	E.1.32
4.3 Core Size and Timing	E.1.33
4.4 Error Stops	E.1.33

JOSHUA DPSTM-500	Title Module TRAMP	Vol. IV	Page E.1.2
		Rev.	Date 6/70

(Blank)

1. Theory

1.1 Introduction

The TRAMP module computes transport matrices for each zone and energy group in a supercell. These matrices are blocked by zone and energy band and written on the Lattice Intermediate Data Set (LIDS).

Two techniques for computing transport matrices have been implemented in this version of TRAMP. The first is the "direct integration" method using the formalism of Carlvik⁽¹⁾. This method is used for both one- and two-dimensional geometries. The second technique is the "cosine currents" method⁽²⁾ which can be applied only to one-dimensional annular geometry.

1.2 Transport Equation

We consider one zone of a supercell and one neutron energy group. The zone consists of sub-regions and faces which bound the zone. Let index n (or m) number this collection of sub-region and faces whose volumes and surfaces are $V(m)$ and $S(m)$ respectively. If there is a uniform isotropic source density, $H(m)$, in sub-region m , the total source is $H(m)V(m)$. This volume source produces a flux, $\phi(n)$, in sub-region n , and, by definition, this flux is given by the product of the total source times the transport matrix element, $T^{VV}(n,m)$. The superscript vv denotes a volume-volume matrix element.

The volume source in sub-region m also produces outward directed currents, $J^{out}(n)$, which pass through faces of the zone. These currents are the product of the total source and the transport matrix element, $T^{SV}(n,m)$. The superscript sv denotes a surface-volume matrix element.

There are also neutrons entering the zone through the faces from other zones. These sources are the inward directed current densities, $J^{in}(m)$, and the total currents are $J^{in}(m)S(m)$. These surface sources produce fluxes in sub-regions and outward directed currents. The corresponding transport matrix elements are $T^{VS}(n,m)$ and $T^{SS}(n,m)$. We can visualize these elements as being arranged in a matrix as shown in Figure 1.

- (1) I. Carlvik, Proc. Third Int. Conf. Peaceful Uses of Atomic Energy, Volume 2, page 265, (1965)
- (2) A. Müller and E. Linnartz, Nukleonik 5, 1 (1963).

Figure 1. Schematic of a Transport Matrix for One Zone and Group.

The flux in a sub-region (or outward current through a face) is the sum of the individual contributions from each source. We can then write the Transport Equations as

$$\phi(n) = \sum_m T^{VV}(n,m)H(m)V(m) + \sum_m 4T^{VS}(n,m)J^{in}(m)S(m) \quad (1)$$

$$J^{out}(n) = \sum_m T^{SV}(n,m)H(m)V(m) + \sum_m T^{SS}(n,m)J^{in}(m)S(m) \quad (2)$$

As we will show later, the 4 in the second term of (1) permits a simple symmetry relation to be used. These symmetry relations are:

$$T^{VV}(n,m) = T^{VV}(m,n) \quad (3)$$

$$T^{SV}(n,m) = T^{VS}(m,n) \quad (4)$$

$$T^{SS}(n,m) = T^{SS}(m,n) \quad (5)$$

The transport matrix satisfies neutron conservation conditions. If $\Sigma(n)$ is the total cross section of sub-region n , the total loss by collision in sub-region n is $\Sigma(n)\phi(n)V(n)$. The total loss by escape through face n is $J^{out}(n)S(n)$. Multiply (1) by $\Sigma(n)V(n)$, multiply (2) by $S(n)$, sum over all n , and rearrange terms.

$$\begin{aligned} & \sum_n \Sigma(n)\phi(n)V(n) + \sum_n J^{out}(n)S(n) = \\ & \sum_m H(m)V(m) \left\{ \sum_n \Sigma(n)V(n)T^{VV}(n,m) + \sum_n S(n)T^{SV}(n,m) \right\} \\ & + \sum_m J^{in}(m)S(m) \left\{ \sum_n \Sigma(n)V(n)4T^{VS}(n,m) + \sum_n S(n)T^{SS}(n,m) \right\} \quad (6) \end{aligned}$$

The LHS is the total loss of neutrons. If the bracketed terms are each unity, the RHS is the total source of neutrons and neutrons will be conserved for an arbitrary source distribution. Thus, conservation requires that

$$\sum_n \Sigma(n) V(n) T^{VV}(n, m) + \sum_n S(n) T^{SV}(n, m) = 1 \quad (7)$$

$$\sum_n \Sigma(n) V(n) 4 T^{VS}(n, m) + \sum_n S(n) T^{SS}(n, m) = 1 \quad (8)$$

1.3 Direct Integration Method

We will first derive expressions for the $T^{VV}(n, m)$ elements of the transport matrix and then show how the remaining elements can be computed. The formalism used here follows that of reference (1). For convenience, we will use n and m as subscripts in this section.

Consider an infinite uniform isotropic line source of neutrons parallel to the z -axis (see Figure 2). Let τ be the distance in mean-free-paths measured in the x - y plane from the source. The number of neutrons which are emitted into the solid angle element $\sin \phi \, d\phi \, d\alpha$ is $\sin \phi \, d\phi \, d\alpha / 4\pi$. Of these, the fraction $\exp(-\tau/\sin \phi)$ escape collision. The number of neutrons which reach τ is the sum of contributions from all segments of the line source.

$$\begin{aligned} dN(\tau) &= \frac{2d\alpha}{4\pi} \int_0^{\pi/2} d\phi \sin \phi \exp(-\tau/\sin \phi) \\ &= \frac{d\alpha}{2\pi} K_{12}(\tau) \end{aligned} \quad (9)$$

where the Bickley functions are given by

$$K_{in}(x) = \int_0^{\pi/2} d\phi \sin^{n+1} \phi \exp(-x/\sin \phi) \quad (10)$$

Figure 2. Coordinate System

In Figure 3 we show the x-y plane cutting sub-regions n and m and a u-h coordinate system tilted at an angle α . If the line source is on the trajectory line in sub-region m at coordinate (u,h), the number of neutrons which collide in sub-region n is the number which reach n, $(d\alpha/2\pi)K_{12}[\Sigma_m(u_m-u) + \tau_{mn}]$, minus those which escape n $(d\alpha/2\pi)K_{12}[\Sigma_m(u_m-u) + \tau_{mn} + \tau_n]$. The number of collisions in n from all sources on the trajectory strip of width dh is then

$$\begin{aligned} & \frac{dh d\alpha}{2\pi} \int_0^{u_m} du \{ K_{12}[\Sigma_m(u_m-u) + \tau_{mn}] - K_{12}[\Sigma_m(u_m-u) + \tau_{mn} + \tau_n] \} \\ &= \frac{dh d\alpha}{2\pi \Sigma_m} \{ K_{13}(\tau_{mn}) - K_{13}(\tau_{mn} + \tau_m) - K_{13}(\tau_{mn} + \tau_n) + K_{13}(\tau_{mn} + \tau_n + \tau_m) \} \end{aligned} \quad (11)$$

If we integrate this expression over all trajectory strips dh and starting angles $d\alpha$, we obtain the number of collisions in n owing to a uniform distribution of unit isotropic sources in m. Since the number of collisions in n is equal to $\Sigma_n V_n \phi$, and a uniform unit source in m is equal to a total source of V_m , we can write the expression for the transport matrix element T_{nm}^{VV} .

$$T_{nm}^{VV} = \frac{1}{\Sigma_n V_n \Sigma_m V_m} \int \frac{d\alpha}{2\pi} \int dh P_{nm}^{VV}(\alpha, h) \quad (12)$$

$$\begin{aligned} P_{nm}^{VV}(\alpha, h) &= K_{13}(\tau_{mn}) - K_{13}(\tau_{mn} + \tau_m) - K_{13}(\tau_{mn} + \tau_n) \\ &\quad + K_{13}(\tau_{mn} + \tau_m + \tau_n) \end{aligned} \quad (13)$$

These expressions apply only to $n \neq m$. For $n=m$ expression (11) becomes

$$\frac{dh d\alpha}{2\pi} \int_0^{u_m} du \{ K_{12}[0] - K_{12}[\Sigma_m(u_m-u)] \} \quad (14)$$

The diagonal transport matrix element is then

Figure 3. Geometry of the (α, h) Plane

$$T_{mm}^{vv} = \frac{1}{\Sigma_m V_m} - \frac{1}{\Sigma_m V_m^2} \int \frac{d\alpha}{2\pi} \int dh P_{mm}^{vv}(\alpha, h) \quad (15)$$

$$P_{mm}^{vv}(\alpha, h) = K_{i3}(0) - K_{i3}(\tau_m) \quad (16)$$

The first term in (15) is obtained by noting that

$$\int dh u_m = V_m \quad (17)$$

$$\int \frac{d\alpha}{2\pi} = 1 \quad (18)$$

Next, we consider the case where the receptor n is a surface and source element m is a sub-region. The expression analogous to (11) is

$$\begin{aligned} & \frac{dh d\alpha}{2\pi} \int_0^{u_m} du K_{i2}[\Sigma_m(u_m - u) + \tau_{mn}] \\ &= \frac{dh d\alpha}{2\pi \Sigma_m} \{K_{i3}(\tau_{mn}) - K_{i3}(\tau_{mn} + \tau_m)\} \end{aligned} \quad (19)$$

The total neutrons passing through the face is $S_n J_n^{\text{out}}$ and the uniform unit source in m is equal to a total source of V_m . We can then write the transport matrix element as

$$T_{nm}^{sv} = \frac{1}{S_n \Sigma_m V_m} \int \frac{d\alpha}{2\pi} \int dh P_{nm}^{sv}(\alpha, h) \quad (20)$$

$$P_{nm}^{sv}(\alpha, h) = K_{i3}(\tau_{mn}) - K_{i3}(\tau_{mn} + \tau_m) \quad (21)$$

We have been able to derive the previous expressions because we assumed the volume sources were uniform and isotropic. The remaining transport matrix elements involve surface sources, that is, neutron currents from other zones. We must make some assumption concerning the angular distribution of the currents or the character of the external medium. Let us assume that the medium surrounding the zone is a large homogeneous medium with a uniform isotropic volume source. We can then use equation (11) to compute collisions in n from the sources in m . However, we must base the transport matrix element T_{nm}^{VS} on the total surface source resulting from the external volume source. This total surface source is just

$$\int \frac{d\alpha}{2\pi} \int dh \int_0^{u_m} du K_{12}[\Sigma_m(u_m - u)]$$

$$= \frac{1}{\Sigma_m} \int \frac{d\alpha}{2\pi} \int dh \{K_{13}(0) - K_{13}(\tau_m)\} \quad (22)$$

As we let the size of the external medium become infinite, the bracketed term is just $K_{13}(0) = \pi/4$. In terms of the α and h integration, the surface area can be expressed as

$$S_m = \frac{1}{2} \int d\alpha \int dh \quad (23)$$

so that the total surface source becomes $S_m/4$. We can now write the transport matrix element T_{nm}^{VS} as

$$T_{nm}^{VS} = \frac{1}{\Sigma_n V_n S_m} \int \frac{d\alpha}{2\pi} \int dh P_{nm}^{VS}(\alpha, h) \quad (24)$$

$$P_{nm}^{VS}(\alpha, h) = K_{13}(\tau_{mn}) - K_{13}(\tau_{mn} + \tau_n) \quad (25)$$

We note that the factor of 4 arbitrarily included in (1) cancels the 4 in the total surface source, and that the resulting T_{nm}^{VS} is the same as T_{mn}^{SV} .

JOSHUA DPSTM-500	Title Module TRAMP	Vol. IV	Page
		Rev.	E.1.12 Date 6/70

Finally, when both n and m are surfaces, we combine the arguments leading to (20) and (24) to obtain

$$T_{nm}^{ss} = \frac{4}{S_n S_m} \int \frac{d\alpha}{2\pi} \int dh P_{nm}^{ss}(\alpha, h) \quad (26)$$

$$P_{nm}^{ss} = K_{13}(\tau_{nm}) \quad (27)$$

These expressions for the transport matrix elements are still in terms of a double integral over α and h . For two-dimensional geometries these integrals must be done numerically. For one-dimensional annular geometry it is apparent there is no dependence on α , so that the term $d\alpha/2\pi = 1$, and only the h integration need be done numerically.

1.4 Cosine Currents Method

Approximate transport matrices can be calculated for zones with full annular geometry using the method of cosine currents. This method works well for sub-regions which are 1) thin enough that the flat source approximation is valid, and 2) thick enough that escaping currents have a cosine-like angular distribution. Only a brief sketch of the method will be given here.

Consider an annular sub-region which has an inner radius R_n , an outer radius R_{n+1} , volume V_n , total cross section Σ_n , and a uniform isotopic volume source. The sub-region is characterized by a set of "escape" coefficients and "transmission" coefficients.

- P_n^0 - The fraction of the uniform volume source which escapes through the outer surface at R_{n+1} .
- P_n^1 - The fraction of the uniform volume source which escapes through the inner surface at R_n .
- P_n^{oi} - The fraction of a cosine-shaped current on the outer surface at R_{n+1} which passes through the sub-region and escapes through the inner surface at R_n .
- P_n^{oo} - The fraction of a cosine-shaped current on the outer surface at R_{n+1} which passes through the sub-region and escape through the outer surface at R_{n+1} .

JOSHUA DPSTM-500	Title Module TRAMP	Vol. IV	Page
		Rev.	E.1.13 Date 6/70

P_n^{10} - The fraction of a cosine-shaped current on the inner surface at R_n which passes through the sub-region and escapes through the outer surface at R_{n+1} .

It can be shown that these coefficients are interrelated and that only P_n^0 and P_n^1 need be computed.

$$P_n^{00} = 1 - \alpha_{2n}[1 + \alpha_{3n}(P_n^0 - P_n^1)] \quad (28)$$

$$P_n^{10} = 1 - \alpha_{3n}P_n^1 \quad (29)$$

$$P_n^{01} = \alpha_{2n}P_n^{10} \quad (30)$$

$$\alpha_{1n} = \Sigma_n(R_{n+1} - R_n) \quad (31)$$

$$\alpha_{2n} = R_n / R_{n+1} \quad (32)$$

$$\alpha_{3n} = 4\Sigma_n V_n / S_n \quad (33)$$

The P_n^0 and P_n^1 have been evaluated and expressed as rational functions of α_{1n} and α_{2n} .

Suppose that there is a unit total source in sub-region m and zero source in all other sub-region. The source produces an inward current, C_m^- , at R_m equal to the escape fraction from sub-region m , P_m^1 . This current enters sub-region $m-1$. The number that pass through to R_{m-2} is $C_{m-1}^- = P_{m-1}^{01} C_m^-$. We can continue following the inward current using the relationship

$$C_n^- = P_n^{01} C_{n+1}^- \quad (34)$$

until we reach the innermost sub-region or a face. If there is no inner face, $C_1^- = 0$, and all of the current which escapes sub-region 1 is directed outward.

$$C_2^+ = P_1^{oo} C_2^- \quad (35)$$

The outward currents are computed recursively using

$$C_{n+1}^+ = P_n^{io} C_n^+ + P_n^{oo} C_{n+1}^- (+P_m^o \text{ if } n=m) \quad (36)$$

until we reach the outer surface. When all currents resulting from the unit source in sub-region m have been computed, the fluxes can be computed from

$$\Sigma_n \phi_n V_n = (C_n^+ + C_{n+1}^-) - (C_{n+1}^+ + C_n^-) \quad (37)$$

that is, the total current entering the sub-region minus that leaving the sub-region. By definition this flux is the matrix element $T^{VV}(n,m)$. The remaining matrix elements are computed in a similar manner.

2. Program Logic

2.1 Interfaces

TRAMP reads the following records:

```
( 'TEMP'. 'LIDS'. DLAT. VLAT → V)
'INPUT'. 'TRAMP'
V. 'GRPS'. NCASE. O. O
V. 'CØMP'. NCASE. O. O
V. 'GEØT'. NCASE. O. O
V. 'ZØNE'. NCASE. O. O
V. 'XSEC'. NCASE. NEB. O
```

TRAMP writes the records:

```
V. 'TMAT'. NCASE. NEB. NZN
```

where NEB and NZN are energy band and zone numbers.

2.2 Overall Structure

A flow diagram for the TRAMP module is shown in Figure 4. The basic steps are:

1. Read and store the input records.
2. Compute and store tables of the $K_{13}(x)$ function needed later for fast table lookup.
3. Read a XSEC block for one energy band.
4. Set up the geometry constants and cross sections for one zone and one energy band using subroutine ZNCØN.
5. Establish the (α, h) integration meshes. These may be user defined or will default to $\Delta\alpha = 10^\circ$, $\Delta h = 0.1$ of the smallest dimension in the zone (see subroutine HMESH).
6. Determine the symmetry properties of the zone and set up the skeleton of the TMAT block for one zone and one energy band using subroutine PRØTM. The symmetry is expressed in terms of three constants:

Figure 4. TRAMP Flow Diagram

JOSHUA DPSTM-500	Title Module TRAMP	Vol. IV	Page E.1.17
		Rev.	Date 6/70

LSYM = 1, Full 2-dimensional geometry involving sub-assembly kinds NSAK = 1,3,5,6, or 9.

= 2, Pseudo 2-dimensional geometry involving only sub-assembly kinds NSAK = 2,4, or 8.

NAZ = Number of azimuthally symmetric sectors.

LRT = 1, All elements in an azimuthal sector are symmetric about the azimuthal center of the sector.

= 0, Otherwise.

7. If LSYM = 2 and NAZ = 1 (full annular geometry), the 1-dimensional generator TMG1 can be used. A user specified number describes the method to be used.

METHØD = 1, Direct integration

= 2, Cosine currents

If LSYM = 2 and NAZ > 1, the zone is some combination of full annular, hexagonal, or square geometry. The user can use the "cylindrical cell" approximation (ICYL≠0) and TMG1 will automatically convert these geometries to full annular geometry preserving all volumes.

8. If LSYM = 1, or LSYM = 2, NAZ > 1, ICYL = 0, the 2-dimensional generator TMG2 is used.
9. The TMAT block resulting from either step 7 or 8 is written to the LIDS.
10. Steps 6-9 are repeated until all zones have been done.
11. Steps 3-10 are repeated until all energy bands have been done.

It frequently occurs in mixed lattice calculations that the outermost zones in each cell are identical. Since these are the zones which require the most computing time (because of the azimuthal sub-division), an input table is used to avoid recalculation:

IDØ(N) = 0, Unique zone N, calculate a TMAT block

> 0, Zone N is the same as zone IDØ(N).

2.3 Zone Constants (ZNCØN,MTXS)

Frequently used properties of a zone are precalculated by ZNCØN and placed in a CØMMØN area.

These properties are:

1. Sub-region volumes
2. Face areas
3. Tables relating indices relative to the zone with those relative to the supercell
4. Tables relating materials to sub-regions
5. Macroscopic cross sections by material and the groups in one energy band.

The first four properties can be extracted directly from the ZØNE block. Macroscopic material cross sections are obtained using subroutine MTXS which uses microscopic data from the XSEC block and concentrations from the CØMP block.

Later calculations in the FIST module are performed using the "Transport Approximation" where the transport cross section is used in place of the total cross section. The transport cross section must also be used to compute the transport matrices. The definition of the σ^{tr} on the MULTIGRP data set includes $n2n$ but not absorption. The transport cross section computed by MTXS, σ_{eff}^{tr} , and used to generate the transport matrices, is then

$$\sigma_{eff}^{tr} = (\sigma^{tr} - 2\sigma^{n2n}) + (\sigma^c + \sigma^f + \sigma^{n2n}) \quad (38)$$

where the superscripts denote transport, $n2n$, capture, and fission.

2.4 Symmetry Considerations

It is essential to use the geometric symmetry (if any exists) of the zone to reduce the size of the transport matrix. To illustrate this point, consider an annular zone with 4 regions, inner and outer faces, and 6 azimuthal sub-divisions. There are then 24 sub-regions, 12 faces, and a total of $36 \times 37/2 = 666$ matrix elements are needed. Because of the rotational symmetry of the zone, many of

these elements are the same. In fact, there are only 84 different elements. While the difference between 666 and 84 elements may not seem large, when 84 energy groups are considered the number of elements is 55,944 versus 7,056.

We have not found a practical algorithm which will deduce the general symmetry properties of a zone. Most of the practical cases, however, involve only rotational symmetry and we have developed a simple algorithm for this case.

We will refer to sub-regions and faces as elements of the zone, and transport matrix elements as TMAT elements. Consider two classes of elements; those which cannot lead to repeated TMAT elements, and those which can. In the first class are:

Type 1 - a full circular element centered at the origin (∞ - fold symmetry).

Type 2 - a square element centered at the origin (4 - fold symmetry)

Type 3 - a hexagonal element centered at the origin (6 - fold symmetry)

The elements in the second class are designated by Type 4 and are those which do not include the origin. Elements which do include the origin but are not of types 1,2,3 are considered to have 1-fold symmetry and will destroy any symmetry of the zone.

We define a group as a collection of elements which would be identical if rotated into a standard position. Elements are assigned to groups in the following manner. If an element is of type 1-3, it is placed in a new group. If an element is of type 4, characteristic properties of the element are computed and matched against the characteristic properties of each group previously defined. If a match is found, the element is added to the group. If no match is found, a new group is added.

The trick is to select characteristic properties of an element which are easy to compute and which will properly discriminate between unlike elements. The following list of properties are used:

1. Material number for sub-region, zero for a face.
2. Number of linear sides.
3. Number of radial sides

4. Volume of sub-region or area of face.
5. Radius from origin to center of element.
6. $\langle \theta - \bar{\theta} \rangle$
7. $\langle Rl \rangle$

If R_n , θ_n , l_n denote the radius to the center of a side of an element, the rotation angle to the center of a side of an element, and the length of the side of an element, properties 5 and 6 are given by

$$\langle \theta - \bar{\theta} \rangle = \frac{\sum_n R_n (\theta_n - \bar{\theta})}{\sum_n R_n} \quad (39)$$

$$\bar{\theta} = \frac{1}{N} \sum_n \theta_n \quad (40)$$

$$\langle Rl \rangle = \frac{\sum_n R_n l_n}{N} \quad (41)$$

where the sum is over all N sides of the element. These properties have been found to discriminate between elements encountered in sub-assembly types 2-6,8,9. There is no guarantee they will be generally successful.

The previous steps lead to a set of symmetry groups. If S_g denotes the number of azimuthal sectors required by group g, the S_g are given by:

<u>Element type</u>	<u>S_g</u>
1	∞
2	4
3	6
4	Number of elements in group

The number of sectors required for the zone, S, is the smallest common factor of the set of S_g . Some examples are:

<u>Sg</u>	<u>S</u>
$\infty, 6, 12$	6
$4, 6, 12$	2
$1, 4, 6$	1

The next step is to assign elements to sectors. Let 0 denote an inner sector which contains all elements of types 1-3, and 1, 2, 3, ..., S denote outer angular sectors containing elements of type 4 (see Figure 1). Arbitrarily assign position numbers 1, 2, 3, ..., N_0 to the elements of sector 0, and 1, 2, 3, ..., N_1 to the elements which lie in sector 1. Each element of the remaining sectors will have a corresponding element in sector 1.

We can visualize the transport matrix as a collection of sub-matrices (see Figure 1) if the elements are numbered first by sector and second by position within a sector. Thus in Figure 1, the block labeled ① is an $N_0 \times N_0$ matrix for transport between elements of sector 0 and sector 0. Because of the symmetry properties of the transport matrix, only a half-matrix (shaded portion) is unique and need be stored. If I and J represent initial and final position numbers, and L represents an index to a matrix element, then

$$L = I + (J-1)(N_0-1) - (J-1)(J-2)/2 \quad (42)$$

The total number of elements is

$$L_0 = N_0(N_0+1)/2 \quad (43)$$

The block labeled ② is an $N_0 \times N_1$ matrix for transport between elements of sector 0 and sector 1. Because of the symmetry the same block is used for transport between sector 0 and any non-zero sector. If these elements are stored following those given above, the index L is given by

$$\begin{aligned} L &= L_0 + I + N_0(J-1), \text{ if } I \text{ lies in sector } 0 \\ &= L_0 + J + N_0(I-1), \text{ if } J \text{ lies in sector } 0 \end{aligned} \quad (44)$$

The total number of elements in this block is

$$L_1 = N_0 N_1 \quad (45)$$

INITIAL ELEMENTS

	SECTOR 0	SECTOR 1	SECTOR 2	SECTOR 3	SECTOR 4
SECTOR 0	①	②	②	②	②
SECTOR 1	②	③	④	⑤	⑥
SECTOR 2	②	④	③	④	⑤
SECTOR 3	②	⑤	④	③	④
SECTOR 4	②	⑥	⑤	④	

FINAL ELEMENTS

Figure 1. Schematic of a Transport Matrix

The block labeled ③ is an $N_1 \times N_1$ matrix for transport between elements of sector 1 and sector 1 (or 2 and 2, 3 and 3, etc.). Again, only the half-matrix is unique, so that the index L is given by

$$L = L_0 + L_1 + I + (J-1)(N_1-1) - (J-1)(J-2)/2 \quad (46)$$

The size of this block is

$$L_2 = N_1(N_1+1)/2 \quad (47)$$

The block labeled ④ is an $N_1 \times N_1$ matrix for transport between elements of sector 1 and sector 2 (or 2 and 3, 3 and 4, etc.). The index L is given by

$$L = L_0 + L_1 + L_2 + I + (J-1)(N_1-1) - (J-1)(J-2)/2 \quad (48)$$

and the number of elements in the block is L_2 . The same procedure is continued for the remaining blocks ⑤, ⑥, etc. In this manner we can store only the unique elements (shaded area in Figure 1) of the matrix.

Further compaction is possible in the case where all elements in a sector are symmetric about the azimuthal center of the sector. In Figure 1, block ⑥ is the same as block ④. In general, the number of blocks of type ③, ④, ⑤, etc., that are unique is given by

$$S^1 = S, \text{ if there is no internal symmetry in a sector,} \\ = 1 + \text{integer part of } (S/2), \text{ if there is internal symmetry.}$$

The total number of matrix elements that must be stored is then

$$L_x = L_0 + L_1 + S^1 L_2 \quad (49)$$

2.5 Numerical Integration

The integrals in equation (12) are evaluated numerically using Simpson's Rule for the h -integration, and the mean-value rule for the α -integration. These integrals are of the form

$$I = \int_0^{2\pi} \frac{d\alpha}{2\pi} \int_0^H dh P(\alpha, h) \quad (50)$$

where H is the radius of the smallest circle which will contain the zone. H is obtained with subroutine GETMAX.

If Δh_0 is the desired h-mesh spacing, there would be

$$N_0 = H/\Delta h_0 \quad (51)$$

intervals. Simpson's Rule required an even number of intervals so that we set

$$\begin{aligned} N_1 &= N_0, \text{ if } N_0 \text{ is even} \\ &= N_0+1, \text{ if } N_0 \text{ is odd} \end{aligned} \quad (52)$$

and compute

$$\Delta h_1 = H/N_1 \quad (53)$$

as the actual mesh spacing.

The h mesh points are then

$$h_n = n\Delta h_1, \quad n=0,1,2,\dots,N_1 \quad (54)$$

The associated integration weights are

$$\begin{aligned} W_n^h &= \Delta h_1/3, \quad n=0, N_1 \\ &= 4\Delta h_1/3, \quad n \text{ odd} \\ &= 2\Delta h_1/3, \quad n \text{ even} \end{aligned} \quad (55)$$

If NAZ is the number of equivalent azimuthal sectors, the sector width is $2\pi/\text{NAZ}$. If the sector symmetry test is on (LRT=1), we need to consider only half a sector. The α -integration range is then

$$\begin{aligned} \alpha_{\max} &= 2\pi/\text{NAZ}, \quad \text{if } \text{LRT}=0 \\ &= \pi/\text{NAZ}, \quad \text{if } \text{LRT}=1 \end{aligned} \quad (56)$$

If $\Delta\alpha_0$ is the desired α -mesh spacing, there are

$$M_1 = [\alpha_{\max}/\Delta\alpha_0] + 1 \text{ intervals,} \quad (57)$$

where $[x]$ means the integer part of x . The actual α -mesh spacing is then

$$\Delta\alpha_1 = \alpha_{\max}/M_1 \quad (58)$$

We center the α -mesh in the interval $\Delta\alpha_1$, so that

$$\alpha_m = (m - \frac{1}{2}) \Delta\alpha_1, \quad m=1, 2, \dots, M_1 \quad (59)$$

The integration weights are given by

$$\begin{aligned} W^\alpha &= 1/NAZ, \quad \text{if } LRT=0 \\ &= 2/NAZ, \quad \text{if } LRT=1 \end{aligned} \quad (60)$$

Finally, the integral (50) is written

$$I = \sum_{m=1}^{M_1} W^\alpha \sum_{n=0}^{N_1} W_n^h P(\alpha_m, h_n) \quad (61)$$

2.6 Tracking Routines (TRACK, TRACK1, TRACK2)

The coordinates (α, h) define a line crossing the zone. The line may have several segments in the zone (for example, a zone with a hole in the middle). Along each segment of the line we need to know where the line intersects faces and sub-regions.

A general purpose tracking subroutine developed for use in Monte Carlo programs is used here to determine the intersections of the line with sub-regions and faces. This routine, called TRACK, uses the geometry descriptions in the GEOT block. We will not further describe the TRACK routine. For our purposes here it is sufficient to state that, given a starting point and direction, the TRACK routine returns a table of track lengths for each sub-region crossed before reaching a sub-assembly boundary.

Since a zone may consist of several sub-assemblies, the track length tables for each must be put together to form a composite table for a segment of the line. Subroutine TRACK1 controls the formation of these tables. The steps are:

1. On the first entry to TRACK1, the geometry tables are searched to determine the intersections of the (α, h) line with the zone faces. These intersections are put in order so that each pair represents a segment of the line.
2. A track length table is produced for one segment of the line by placing a starting point on the left-most face of the segment and calling TRACK. If the face terminating the track is not the same as the right-most face of the segment, the TRACK routine is called again and the resulting track length table appended to the previous one.
3. A segment counter is maintained which informs the program calling TRACK1 how many segments remain to be processed.

For simple annular zones, the track length tables are computed by the TRACK2 subroutine to take advantage of the simpler calculation. The table of annular radii needed by TRACK2 is produced by the GETRAD subroutine from the data in the GEOT block. If full hexagonal or square sub-regions are encountered by GETRAD, they are converted to annular sub-regions with the same volume.

2.7 Transport Matrix Generators(TMGl,TMG2)

The transport matrix generators TMG1 and TMG2 are control subroutines for one- and two-dimensional geometries. The required steps are as follows:

1. Select the α (TMG2 only) and h integration meshes.
2. For each (α, h) line generate the table of intersections with the geometry. TMG1 uses TRACK1, TMG2 uses TRACK2 for this operation.
3. Evaluate the function $P(\alpha, y)$ from equations (13), (16), (21), (25), and (27) for each energy group in the energy band, multiply by the integration weight, and add the result into the TMAT block. This step is done by subroutine TXX.

4. Repeat steps 2-3 until all (α, h) lines have been done.
5. Complete the TMAT block by dividing by the appropriate cross section, area, and volume factors in equations (12), (20), (24), and (26) using subroutine FIXTM.

If the method of cosine currents is specified, TMG1 calls the generator CØSJ1 and stores the results in the TMAT block.

2.8 Evaluation of the $P(\alpha, y)$ Functions (TXX)

The subroutine TXX evaluates the functions $P(\alpha, y)$ defined by equations (13), (16), (21), (25), and (27) given:

1. Tables of the intersections of the (α, h) line with the geometry. These data are in the form of a starting and ending face number and tables of line length and sub-region numbers for each sub-region intersected. These data are generated by TMG1 or TMG2 using the tracking routines.
2. Tables of the total cross section for each sub-region in the zone and energy group in the energy band. These data are generated by ZNCØN using MTXS.

The discussion in Section 1.3 showed that we could treat the medium external to the zone as an infinite medium with a uniform isotropic source. Therefore, we add a 10 mfp thick sub-region to the beginning and end of the (α, h) line across the zone and use only equations (13) for $P^{VV}(\alpha, h)$. The operation of TXX is as follows:

1. Set up a table of mfp distances for one energy group starting with a 10 mfp extra sub-region at the beginning of the line and adding another one at the end of the line. These are the variables τ_n .
2. Loop over all m and $n \geq m$ (upper half of the symmetric matrix).
3. If $n = m$, subtract (16) times the integration weight from the cumulative result in the TMAT block.
4. If $n > m$, add (13) times the integration weight to the cumulative result in the TMAT block.
5. Repeat all steps until all groups in the band have been done.

Since (13) is symmetric in n and m , we need calculate only half of the matrix. The subtraction of the diagonal term in step 3 assures conservation since this term is equal to the sum of all terms calculated in step 4. The resulting TMAT block will sum to zero and the first term in (15) will be added by subroutine FIXTM to obtain the complete TMAT block.

The calculation of the intersections by the tracking routines is time consuming. However, using the same table of intersections for many groups reduces the number of entries to the tracking routine. The result is that, in an actual problem, only 3% of the time was used by the tracking routines, 88% by subroutine TXX, and 9% by all other routines. Because of the importance of optimizing the TXX routine, table look-up with linear interpolation is used to calculate the $K_{13}(x)$ functions.

2.9 Final Corrections to the Matrices (FIXTM)

Following the α and h integrations of the $P(\alpha, h)$ functions, matrices must be divided by the appropriate factors and normalization terms added to the diagonal of the matrix. These factors are:

<u>Term</u>	<u>Scale Factor</u>	<u>Normalization Constant</u>
volume+volume	$1/\sum_n V_n \sum_m V_m$	$1/\sum_m V_m$
volume+surface	$1/\sum_n V_n S_m$	-
surface+volume	$1/S_n \sum_m V_m$	-
surface+surface	$4/S_n S_m$	$1/S_n$

3. Coding Details

The TRAMP module is divided into the following parts:

1. COMMON/DATA/
2. COMMON/SPECS/
3. COMMON/REFPNT/
4. COMMON/TRAK/
5. COMMON/TEMP/
6. COMMON/SCRACH/
7. Main program TRAMP
8. Subroutine HMESH
9. Subroutine GETMAX
10. Subroutine GETRAD

11. Subroutine CØSJ1
12. Subroutine PRØTM
13. Subroutine TMG1
14. Subroutine TMG2
15. Subroutine TXX
16. Subroutine TRACK1
17. Subroutine TRACK2
18. Subroutine TRACK
19. Subroutine ZNCØN
20. Subroutine ESCAPE
21. Subroutine MTXS
22. Subroutine FIXTM
23. Subroutine MANAGE (See manuscript B.1)

3.1 CØMMØN/DATA/

The LIDS blocks NAME, CØMP, GEØT, ZØNE, XSEC, and TMAT, as well as the block KI3F used for interpolation of the K13 functions, are packed into the array defined by

CØMMØN/DATA/DATA(20000).

Refer to manuscript B.1 for a description of this array and its usage.

3.2 CØMMØN/SPECS/

This area contains the data:

TDSN - Temporary data set name

DLAT - Lattice name

VLAT - Lattice version

NCASE - Case number

3.3 CØMMØN/REFPNT/

This area passes the origins of the LIDS blocks to the subroutines which use them. The origins used are:

KGØRG - GEØM origin

KCØRG - CØMP origin

KZØRG - ZØNE origin

KXØRG - XSEC origin

KTØRG - TMAT origin
 KI3 - KI3F origin
 KZN - pointer to current zone data

3.4 CØMMØN/TRAK/

This area is used to pass tracking information to and from the subroutines TRACK and TRACK1. The items in this area are as follows:

NLP - Lattice position number
 NFC - Face number
 NSR - Subregion number
 XO - x-coordinate of starting position of neutron
 YO - y-coordinate of starting position of neutron
 ZO - z-coordinate of starting position of neutron
 THO - Track angle θ with x-y plane measured from x axis
 STHO - $\sin(\theta)$
 CTHO - $\cos(\theta)$
 PHO - angle ϕ measured from z axis to the track.
 (set to $\pi/2$ in TRAMP)
 SPHO - $\sin(\phi)$
 CPHO - $\cos(\phi)$
 KGØX - Total number of track lengths in subassembly NSA
 KGØXX - Maximum number of track lengths allowed
 NSRT(KGØ) - Subregion that contains track KGØ
 DXT(KGØ) - Length of track KGØ, KGØ = 1, KGØX

3.5 CØMMØN/TEMP/

This common area is used to pass current group, energy band, and zone numbers; the maximum values of certain variables, and tables used in calculating collision probabilities. The items in this array are as follows:

IL - Lower group index for current energy band
 IH - Upper group index for current energy band
 NEB - Current energy band number
 NZN - Current zone number
 ISRX - Number of subregions in zone NZN
 ISRXX - Maximum number of subregions allowed in one zone

IFCX - Number of faces in zone NZN

IFCXX - Maximum number of faces allowed in one zone

ISFX - Number of subregions plus faces in zone NZN

ISFXX - Maximum number of subregions plus faces in zone

IRDXX - Number of radii in zone NZN

IRDXX - Maximum number of radii allowed in one zone

NGTX - Number of track segments in zone NZN

NGTXX - Maximum number of track segments allowed in one zone

IFC1 - Face where neutron enters zone NZN

IFC2 - Face where neutron leaves zone NZN

NBINCX - Number of entries in Bickly tables

BINC - Increment used in Bickly tables

IMTXX - Maximum number of materials allowed in one zone

INSRT(NSR) - Number of sub-region NSR, relative to the zone NZN

INFCT(NFC) - Number of face NFC, relative to the zone NZN

VSR(ISR) - Volume of subregion ISR

AFC(IFC) - Area of face IFC

IMTSR(ISR) - Relative number of the material in sub-region ISR

RAD(IRD) - Table of radii for zone NZN

DX(N) - Table of track lengths in zone NZN
N=1, NGTX

ISRT(N) - The relative number of the sub-region that contains track N.

SIGT(IMT,I)- Transport cross section of material IMT for group I

NELT(ISF,JSF)-Table relating element numbers to indices ISF and JSF

Note that indices starting with I or J number items within a zone, and indices starting with N number geometry items in the supercell.

3.6 COMMON/SCRACH/

This area is used by subroutines CØSJ1, PRØTM, TXX, TRACK, and ZNCØN as storage for temporary arrays.

4. User Information

4.1 Input

TRAMP cannot be executed as a stand-alone module since it requires the NAME, COMP, GEOT, and ZONE blocks created by ASP and LAGER, and the XSEC and GRPS block created by CRISP. If these modules have been executed, then TRAMP needs only the record:

Name: 'INPUT'. 'TRAMP'

List: DLAT, VLAT, TDSN, NCASE, LPRNT, ICYL, NZNX, (IDØ(N),
METH(N), ICYL(N), DHT(N), DAT(N), N=1, NZNX)

DLAT - Lattice name

VLAT - Lattice version

TDSN - Temporary data set name = 'TEMP'

NCASE - Case number

LPRNT - If LPRNT=1, print the maximum number of words used in the DATA array

ICYL(N) - Cylindrical cell flag
ICYL = 1 - cylindrical cell approximation
= 0 - no cylindrical cell approximation

NZNX - Number of zones

IDØ(N) - If IDØ(N)= 0, then collision probabilities for zone N are not calculated.

METH(N) - Method used for creating collision probabilities in zone N.

METH(N) = 1 Direct integration
= 2 Cosine currents
= 3 Statistical integration

DHT(N) - H-integration mesh spacing for zone N

DAT(N) - Angular integration mesh spacing for zone N

Normally, this record is created by the RAHAB control module. These items are used to establish the names of the input and output records.

4.2 Output

For each energy band NEB and zone NZN, TRAMP writes the record 'TMAT'.CASE.NEB.NZN on the LIDS.

JOSHUA DPSTM-500	Title Module TRAMP	Vol. IV	Page
		Rev.	E.1.33 Date 6/70

4.3 Core Size and Timing

The size of the various parts of TRAMP are given in Table I. Timings are also given for a one cell problem with 84 groups, 2 zones, 28 sub-regions using the cylindrical cell approximation and direct integration.

4.4 Error Stops

All stops are handled by the JØSHUA Operating System and are summarized in Table II.

TABLE I

CORE SIZE AND TIMINGS FOR TRAMP MODULE
(Size in bytes, time in CPU seconds.)

Subprogram	Size	Total Time	Average Time	Entries
MAIN	3,560	.24960	.24960	1
HMESH	798	.01664	.00416	4
GETMAX	788	.01664	.00277	6
GETRAD	1,902	-	-	2
PRØTM	8,222	2.87872	.71968	4
TMG1	1,942	.01664	.00832	2
TMG2	1,088	.11648	.05824	2
TXX	1,602	75.22944	.48224	156
TRACK1	2,542	.28288	.00205	138
TRACK2	1,080	.03328	.00128	26
TRACK	5,5856	1.16480	.00896	130
ZNCØN	1,468	.03328	.00832	4
MTXS	942	.04992	.00624	8
CØSJ1	1,768			
ESCAPE	1,478			
FIXTM	1,994	2.46272	.61568	4
MANAGE	11,840*	2.17984**	.01689	129
BICK	4,290			
IHCSLØG	442			
IHCSSCN	493			
IHCLEXP	642			
IHCLSQRT	322			
IHCSEXP	432			
IHCSSQRT	329			
/DATA/	40,000			
/SPECS/	28			
/REFPNT/	28			
/TRAK/	1,656			
/TEMP/	43,276			
/SCRACH/	11,244			
/DATCAT/	2,440			
JØS	75,144***			
TOTALS	229,636	84.73008		616

* Includes all routines of data manager.

** Includes time for JOSHUA Operating System and output routines.

*** Includes JOSHUA Operating System and buffers.

JOSHUA
DPSTM-500

Title Module TRAMP

Vol. IV	Page E.1.35
Rev.	Date 6/70

(Blank)

TABLE II

ERROR STOPS AND MESSAGES

Message/Explanation

TRAMP.MAIN,NAME,COMP,GEOT,ZONE BLOCK MISSING.
One of the above records is not on the LIDS.

TRAMP.MAIN,XSEC BLOCK MISSING
Cross section block not on the LIDS.

TRAMP.MAIN,CONTROL RECORD MISSING
The record 'INPUT'.'TRAMP' is missing.

TRAMP.MAIN,GRØUPS RECORD MISSING.
The group structure record is not in the LIDS.

TRAMP.MAIN, TØØ MANY SUBREGIONS.
The number of subregions exceeds 200.

TRAMP.MAIN, TØØ MANY FACES.
The number of faces exceeds 100.

TRAMP.GETRAD, SUB-REGIONS OUT OF ØRDER.
Programming error

TRAMP.GETRAD,IMPRØPER NSAK.
Programming error. GETRAD handles only annular, hex, and square geometry.

TRAMP.GETRAD, ERRØR IN GEØMETRY TABLES.
Number of radii in zone not consistent with ZØNE table.

TRAMP.PRØTM,IMPRØPER SYMMETRY TYPE.
Symmetry is none of the types described in section 3.11.

TRAMP.PRØTM, DATA ARRAY ØVERFLØW.
The array DATA (see section 3.1) cannot hold all the data.

TRAMP.TMG2, ILLEGAL METHØD.
The method specified for calculating collision probabilities is other than those described in section 3.13, or has not yet been implemented.

TRAMP.TRACK1, GEØT ERRØR.
The GEØT block is in error.

Work Done By

C. E. Henderson
H. C. Honeck

Report Written By

C. E. Henderson
H. C. Honeck

Contents

	<u>Page</u>
1. Theory	E.2.3
1.1 Introduction	E.2.3
1.2 Spatial Transport	E.2.3
1.3 Volume Sources	E.2.5
1.4 Resonance Reactions	E.2.7
1.5 The (n,2n) Reaction	E.2.8
1.6 The Transport Approximation	E.2.8
1.7 Neutron Balance	E.2.9
2. Program Logic	E.2.10
2.1 Interfaces	E.2.10
2.2 Calculation Sequence	E.2.10
2.3 Epithermal Flux Calculation (EFLUX)	E.2.12
2.4 Thermal Flux Calculation (TFLUX)	E.2.14
2.5 Fast Flux Calculation (FFLUX)	E.2.16
2.6 Fission Source Calculation (FISØR)	E.2.17
2.7 Spatial Flux Calculation (GRITER)	E.2.18
2.8 Group Constants Calculation (GRPCØN)	E.2.19
2.9 Isotope Data Retrieval (ISØDAT)	E.2.20
2.10 Resonance Rates (RESSR)	E.2.21
3. Coding Details	E.2.22
3.1 Module Structure	E.2.22
3.2 Subroutine GRITER	E.2.23
3.3 Subroutine ISØDAT	E.2.24
3.4 CØMMØN/DATA/	E.2.26
3.5 CØMMØN/SPECS/	E.2.27
3.6 CØMMØN/GRØUPS/	E.2.28
3.7 CØMMØN/TEMP/	E.2.29
3.8 CØMMØN/ITSTAT/	E.2.31
4. User Information	E.2.32
4.1 Input	E.2.32
4.2 Output	E.2.33
4.3 Core Size and Timings	E.2.33
4.4 Error Stops	E.2.33

JOSHUA DPSTM-500	Title Integral Transport Modules - Module FIST	Vol. IV	Page E.2.2
		Rev.	Date 6/70

(Blank)

1. Theory

1.1 Introduction

The FIST module solves the multigroup multiregion matrix transport equation to obtain the fine-structure flux in a supercell. Other modules have organized the required data; geometry and composition data (LAGER), resonance data (RESIN), cross sections and scattering matrices (CRISP), and transport matrices (TRAMP). These data are read by FIST and the matrix equations solved by iterative procedures. The resulting fine-structure fluxes, currents, and slowing-down densities are output for use by later editing modules.

The equations solved by FIST are given in the remainder of this section. The following section describes the method of solution and program logic.

1.2 Spatial Transport

Consider one zone of the supercell. The zone consists of a collection of sub-regions and faces which bound the zone. Suppose there is an average source density $H(m,j)$ in the j^{th} group and m^{th} sub-region. If $V(m)$ is the volume of the sub-region, the total source is $H(m,j)V(m)$. This source is referred to as a "volume" source. The neutrons from this source either collide with atoms in this or other sub-regions or escape the zone through a face. Collisions are considered to remove the neutrons from the system, but some will be re-introduced later into the source term.

Another source of neutrons are those which enter the zone through faces, called "surface" sources. If $J^{\text{in}}(m,j)$ is the current density in the j^{th} group entering the zone through the m^{th} face, and $S(m)$ is the surface area, the total surface source is $J^{\text{in}}(m,j) S(m)$. These neutrons will either collide with atoms in sub-regions or escape the zone through a face.

The transport matrix is defined as the average flux in a sub-region (or outgoing current density through a face) resulting from a unit source in another sub-region (or inward current through a face). The elements of the matrix are denoted by $T^{\text{VV}}(n,m,j)$, $T^{\text{VS}}(n,m,j)$, $T^{\text{SV}}(n,m,j)$, and $T^{\text{SS}}(n,m,j)$. The arguments (n,m,j) refer to the n^{th} receptor element, m^{th} source element, and j^{th} group. The superscripts VV, VS, SV, and SS denote volume $\leftarrow$ surface, surface $\leftarrow$ volume, and surface $\leftarrow$ surface terms. The indices n,m are selected so that the smaller values number sub-regions and the larger values number faces.

The flux in the n^{th} sub-region is a summation of the contributions from volume sources in all sub-regions plus contributions from inward currents through the faces.

$$\begin{aligned} \phi(n,j) = & \sum_m T^{VV}(n,m,j) H(m,j) V(m) \\ & + \sum_m 4T^{VS}(n,m,j) J^{\text{in}}(m,j) S(m) \end{aligned}$$

(The factor of 4 is included so that the transport matrix will be symmetric.). Similarly, the outward current is a summation of contributions from volume and surface sources.

$$\begin{aligned} J^{\text{out}}(n,j) = & \sum_m T^{SV}(n,m,j) H(m,j) V(m) \\ & + \sum_m T^{SS}(n,m,j) J^{\text{in}}(m,j) S(m) \end{aligned} \quad (2)$$

Neutron conservation conditions are obtained as follows. Let $\Sigma(n,j)$ be the total cross section in group j and sub-region n . The total number of collisions in group j and sub-region n is then $\Sigma(n,j) \phi(n,j) V(n)$. The number of neutrons which leave the zone through face n and group j is $J^{\text{out}}(n,j) S(n)$. If we multiply (1) by $\Sigma(n,j) V(n)$, multiply (2) by $S(n)$, and sum over all n we obtain after some rearrangement:

$$\begin{aligned} \sum_n \Sigma(n,j) \phi(n,j) V(n) + \sum_n J^{\text{out}}(n,j) S(n) = & \quad (3) \\ \sum_m H(m,j) V(m) \left\{ \sum_n \Sigma(n,j) V(n) T^{VV}(n,m,j) + \sum_n S(n) T^{SV}(n,m,j) \right\} \\ + \sum_m J^{\text{in}}(m,j) S(m) \left\{ \sum_n \Sigma(n,j) V(n) 4T^{VS}(n,m,j) + \sum_n S(n) T^{SS}(n,m,j) \right\} \end{aligned}$$

The LHS is the total loss by collision and escape. If the number of particles is to be conserved, the total loss must equal the total source for an arbitrary source distribution. The necessary conservation conditions are then that

$$\sum_n \Sigma(n,j) V(n) T^{VV}(n,m,j) + \sum_n S(n) T^{SV}(n,m,j) = 1 \quad (4)$$

$$\sum_n \Sigma(n,j) V(n) 4T^{VS}(n,m,j) + \sum_n S(n) T^{SS}(n,m,j) = 1$$

When neutrons escape one zone they enter an adjacent zone. Thus, the inward currents through a face are related to the outward currents through other faces. Let N or M number faces in the supercell. The relationship between inward and outward currents is given by the boundary condition matrix, B(N,M), such that

$$J^{in}(N,j) = \sum_M B(N,M) J^{out}(M,j) \quad (5)$$

If there are no neutrons lost while crossing faces (no vacuum boundaries), conservation conditions are obtained by multiplying (5) by S(N) and summing over all N.

$$\sum_N S(N) J^{in}(N,j) = \sum_M J^{out}(M,j) S(M) \sum_N S(N) B(N,M) / S(M) \quad (6)$$

If the total inward current is to equal the total outward current,

$$\sum_N S(N) B(N,M) / S(M) = 1 \quad (7)$$

1.3 Volume Sources

The volume source term H(m,j) can be decomposed into three contributions.

$$H(m,j) = H_1(m,j) + \rho(c,j) H_2(m,j) + H_3(m,j) \quad (8)$$

The first term is the fission source. Define the fractional fission neutron density produced by isotope k in sub-region m as

$$F_k(m) = \frac{1}{k_{eff}} \left\{ \sum_i \Delta E(i) v_k(i) \Sigma_k^f(m, i) \phi(m, i) V(m) \right. \\ \left. + \text{resonance fission production} \right\} \quad (9)$$

such that

$$\sum_k \sum_m F_k(m) = 1 \quad (10)$$

The eigenvalue k_{eff} is discussed in the next section. The fission source term is then

$$H_1(m, j) = \sum_k \chi_k(j) F_k(m) / V(m) \quad (11)$$

where $\chi_k(j)$ is the normalized fission spectrum for isotope k . The total fission source is

$$\sum_m \sum_j V(m) \Delta E(j) H_1(m, j) = \sum_k \sum_m F_k(m) \sum_j \Delta E(j) \chi_k(j) = 1 \quad (12)$$

as a result of the normalization of the fission spectrum and fission neutron density.

The second term in (8) is the net slowing-down source and consists of the total slowing-down source

$$H_2(m, j) = \sum_{i \neq j} \Delta E(i) \Sigma^S(m, j \leftarrow i) \phi(m, i) \quad (13)$$

times a resonance reduction factor $\rho(c, j)$ discussed in the next section. The term $\Sigma^S(m, j \leftarrow i)$ is the scattering matrix element from group i to group j in sub-region m .

The third term in (8) is the in-group scattering source

$$H_3(m, j) = \Delta E(j) \Sigma^S(m, j \leftarrow j) \phi(m, j) \quad (14)$$

The product $\Delta E(j) \Sigma^S(m, j \leftarrow j)$ is called the "in-group" cross section. We note for later use that the scattering matrix satisfies the normalization condition

$$\Sigma^S(m, i) = \sum_j \Delta E(j) \Sigma^S(m, j \leftarrow i) \quad (15)$$

where $\Sigma^s(m,i)$ is the scattering cross section.

1.4 Resonance Reactions

Resonance capture and fission is treated in the following manner. Let $R_k^a(m,j)$ and $R_k^f(m,j)$ be the resonance absorption and fission rates in isotope k , sub-region m , group j , based on a unit uniform slowing-down density past energy $E(j)$ in the cell c containing sub-region m . The slowing-down density past energy $E(j)$ in cell c is given by

$$Q(c,j) = \sum_{m \in c} V(m) \sum_{i < j} \Delta E(i) \phi(m,i) \sum_{j' \geq j} \Delta E(j') \Sigma_s(m,j' \leftarrow i) \quad (16)$$

The calculation of the rates $R_k^a(m,j)$ and $R_k^f(m,j)$ assumes that the slowing-down density is uniform across the cell. A first-order correction can be applied to improve this approximation since some knowledge of the shape across the cell is available in the fine-structure calculation. This shape factor is taken to be proportional to flux in the preceding group.

$$f(m,j) = \phi(m,j-1) \sum_{m \in c} V(m) / \sum_{m \in c} V(m) \phi(m,j-1) \quad (17)$$

Corrected resonance rates based on a unit slowing-down density are then defined by

$$R_k^{*a}(m,j) = R_k^a(m,j) f(m,j) \quad (18)$$

$$R_k^{*f}(m,j) = R_k^f(m,j) f(m,j) \quad (19)$$

The resonance escape probability for cell c and group j can be defined by

$$\rho(c,j) = 1 - \sum_k \sum_{m \in c} R_k^{*a}(m,j) \quad (20)$$

If $[1 - \rho(c,j)]$ is the fraction of the slowing-down density absorbed in resonances in cell c and group j , then $[1 - \rho(c,j)]Q(c,j)$ is the total number absorbed. These are deducted from the slowing-down source into group j by using the reduction factor

$$\rho(c,j) = 1 - [1 - \rho(c,j)]Q(c,j) / \sum_{m \in c} V(m)H_2(m,j) \quad (21)$$

We can also complete equation (9)

(22)

$$F_k(m) = \frac{1}{k_{eff}} \sum_i v_k(i) \left\{ \Sigma_k^f(m,i) \phi(m,i) \Delta E(i) V(m) + R_k^{*f}(m,i) Q(c,i) \right\}$$

1.5 The (n,2n) Reaction

The (n,2n) reaction could be treated separately, as a fission-like process, or as a scattering-like process. We choose to treat it as a scattering-like process so that it does not distort the definition of the eigenvalue, k_{eff} , and so that we do not need to keep a separate (n,2n) scattering matrix.

Scattering matrices on the MULTIGRP Data Set include the (n,2n) matrix.

$$\sigma_k^s(j \leftarrow i) = \sigma_k^{el}(j \leftarrow i) + \sigma_k^{in}(j \leftarrow i) + 2 \sigma_k^{n2n}(j \leftarrow i) \quad (23)$$

where the superscripts denote elastic, inelastic, and (n,2n). To compensate for the added (n,2n) production and retain the form of the preceding equations, we reduce the value of the absorption cross section and use the effective cross section

$$\sigma_k^{*a}(i) = \sigma_k^a(i) - 2 \sigma_k^{n2n}(i) \quad (24)$$

throughout the calculations.

1.6 The Transport Approximation

It has been found that the assumption of scattering isotropy used to derive the integral transport equation can be greatly improved if we reduce the scattering cross section by the P_1 component of the scattering cross section. This is the "Transport Approximation" and involves three changes:

1. Replace $\sigma_k^t(i)$ by $\sigma_k^{tr}(i)$ throughout, and,
2. reduce the diagonal of the scattering matrix $\sigma_k^s(i \leftarrow i)$ by the amount $[\sigma_k^t(i) - \sigma_k^{tr}(i)]/\Delta E(i)$.
3. Use $\sigma_k^{tr}(i)$ in place of $\sigma_k^t(i)$ in generating the transport matrices so that the normalization equations (4) are valid.

1.7 Neutron Balance

If we use the preceding definitions and compute total reaction rates for the supercell, we would obtain the following equations for absorption, leakage, and production.

$$\text{Absorption} = \sum_m \sum_i \mu(c) \left\{ \Sigma^a(m,i) \phi(m,i) V(m) \Delta E(i) + \sum_k R_k^{*a}(m,i) Q(c,i) \right\} \quad (25)$$

$$\text{Leakage} = \sum_N \sum_i \mu(c) \left[J^{\text{out}}(N,i) - J^{\text{in}}(N,i) \right] S(N) \Delta E(i) \quad (26)$$

$$\text{Production} = \sum_m \sum_i \mu(c) \sum_k \nu_k(i) \left\{ R_k^{*f}(m,i) Q(c,i) + \Sigma_k^f(m,i) \phi(m,i) V(m) \Delta E(i) \right\} \quad (27)$$

where $\mu(c)$ is the number of times cell c occurs in the supercell.

We would find, in general, that absorption + leakage did not equal production, that is, the supercell is not exactly "critical". We introduce an eigenvalue, k_{eff} , which reduces the production so as to obtain a balance.

$$\text{Absorption} + \text{Leakage} = \text{Production}/k_{\text{eff}} \quad (28)$$

or

$$k_{\text{eff}} = \text{Production}/(\text{Absorption} + \text{Leakage}) \quad (29)$$

2. Program Logic

2.1 Interfaces

The FIST module reads the following records:

```
'INPUT'.'FIST'
('TEMP'.'LIDS'.DLAT.VLAT → V)
V.'COMP'.NCASE. 0 . 0
V.'GEOM'.NCASE. 0 . 0
V.'ZONE'.NCASE. 0 . 0
V.'RESD'.NCASE. 0 . 0
V.'CHI' .NCASE. 0 . 0
V.'XSEC'.NCASE.NEB. 0
V.'SMAT'.NCASE.NEB. 0
V.'TMAT'.NCASE.NEB.NZN
```

The FIST module writes the following records:

```
V.'FLUX'.NCASE. 0 . 0
V.'FLUX'.NCASE. 1 . 0
V.'JIN' .NCASE. 0 . 0
V.'JOUT'.NCASE. 0 . 0
V.'QCEL'.NCASE. 0 . 0
V.'RESD'.NCASE. 1 . 0
```

2.2 Calculation Sequence

The present version of FIST is designed to calculate the eigenvalue, k_{eff} , of an active lattice or the spectrum from a uniform fission source in a dead lattice. We will discuss the k_{eff} calculation first.

Two types of iteration are required to solve the integral transport matrix equation. The first type, called "inner" iteration, is used to compute the spatial shape for one energy group with fixed source terms from other energy groups. The second type, called "outer" iteration, is used to settle the thermal flux energy distribution and fission source shape.

At the start of the calculation the spatial fission distribution is not known. We can assume, however, that the spatial distribution of the flux at kev energies is reasonably flat

and start our solution there. We can then compute the epithermal flux, the thermal flux, and the fissions which occur in these energy ranges. Since we know that in thermal reactors the bulk of the fissions occur in the thermal and epithermal ranges, we can calculate a fission source and the fast flux. A few outer iterations on the fast flux should settle the fast fission contributions. It is possible that this fission source yields a flux at kev energies which is sufficiently flat spatially (our starting assumption) so that we could end the calculation at this point. However, the recalculation of the epithermal and thermal fluxes based on the new fission source requires little additional time and is automatically done.

Before describing the calculation sequence in more detail, we will define the terms "fast", "thermal", and "epithermal" energy ranges used above. The energy group structure has associated with it, two numbers defined by:

- JCHI - The last group (highest number, lowest energy) in which the fission spectrum for some isotope is non-zero.
- JFTG - The first group (lowest number, highest energy) in which thermal up-scattering can occur.

If I is a group number, and NEG is the number of groups, the energy regions are defined by:

- Fast $1 \leq I \leq JCHI$
- Epithermal $JCHI < I < JFTG$
- Thermal $JFTG \leq I \leq NEG$

We summarize the calculation sequence contained in the FIST control program as follows:

1. Read the input blocks COMP, GEOM, ZONE, RESD, and CHI from the LIDS. If there is a CHI block, the lattice is active. If there is no CHI block, the lattice is dead and steps 3 - 5 below are omitted.
2. Attempt to read old FLUX, JIN, and JOUT blocks from the LIDS. If no old blocks exist, allocate space in the DATA array for these blocks.
3. Establish a spatially uniform 1/E flux in the fast energy range.

4. Calculate the epithermal flux and fissions using sub-routine EFLUX.
5. Calculate the thermal flux and fissions using sub-routine TFLUX.
6. Calculate the fast flux and fissions using subroutine FFLUX.
7. Recalculate the epithermal flux and fissions using subroutine EFLUX.
8. Recalculate the thermal flux and fissions using sub-routine TFLUX.
9. Rebalance fission production and normalize all results to unit total fission production in the supercell.
10. Write the output blocks FLUX, JIN, JOUT, and QCEL.
11. Correct the resonance rates according to equations (18 - 19) and write the results as record 'TEMP'.'LIDS'.DLAT.VLAT.'RESD'.NCASE.1.0
12. Compute the flux by region and write the results as record 'TEMP'.'LIDS'.DLAT.VLAT.'FLUX'.NCASE.1.0

2.3 Epithermal Flux Calculation (EFLUX)

In the epithermal range neutrons can only lose energy in a scattering event. Therefore the flux in a group depends only on the flux in preceding (higher energy) groups. The calculation starts at the smallest numbered group, $I=JCHI+1$, and proceeds through the largest group $I=JFTG-1$. The steps are as follows:

1. Clear epithermal fission production tables.
2. Repeat the following steps for each group.
3. Obtain the cross sections and slowing-down source for this group and each sub-region using subroutine GRPCON.
4. Obtain resonance rates for this group using subroutine RESSR.
5. Calculate and store resonance fission production from equation (22).
6. Deduct resonance absorption from the slowing-down source into this group according to equation (21).

JOSHUA DPSTM-500	Title Module FIST	Vol. IV	Page E.2.13
		Rev.	Date 6/70

7. Calculate the spatial flux in this group using sub-routine GRITER. If good fluxes and currents are available, use them as a starting guess. If not, use the flux and currents from the previous group.
8. Calculate and store smooth fission production using equation (22), total absorption using equation (25), and total leakage using equation (26).
9. Compute the slowing-down density across the lower energy boundary of this group using equation (16).

JOSHUA DPSTM-500	Title Module FIST	Vol. IV	Page E.2.14
		Rev.	Date 6/70

2.4 Thermal Flux Calculation (TFLUX)

In the thermal energy range neutrons can both gain and lose energy in a scattering event. Therefore, we must iterate both on the spatial shape (inner iteration) and energy shape (outer iteration). The inner iteration is done in subroutine GRITER and the outer iteration is done in TFLUX. The steps are as follows:

1. If thermal fluxes and currents are available from a previous calculation, they are used as the starting guess. If not, compute a starting guess which are the fluxes and currents from the last epithermal group multiplied by a Maxwellian distribution at a temperature of 0.03 ev.
2. Compute the source of neutrons in each thermal group resulting from scatterings in the epithermal and fast range using subroutine GRPCØN. This is called the epithermal source and is constant throughout the outer iterations.
3. Compute the total absorption and leakage implied by the flux and current guess established in step 1.
4. Scale the flux and current guess so that the total absorption and leakage equals the total epithermal source.
5. Start one outer iteration.
6. Repeat the next four steps for each thermal group starting with the highest energy group.
7. Calculate the source into this group from other thermal groups using subroutine GRPCØN, and add the epithermal source.
8. Perform a few (2 - 5) inner iterations on the spatial shape using subroutine GRITER.
9. Compute the spatially integrated flux for this group and save for later convergence test.
10. Accumulate total absorption and leakage using equations (25) and (26).
11. After all thermal groups have been done, rebalance thermal fluxes, currents, and spatially integrated fluxes, so that total absorption plus leakage equals the total epithermal source.

12. Compare the spatially integrated fluxes from this iteration with those from the preceding iteration. If the fractional change in any group is greater than 10^{-5} , do another outer iteration starting at step 5. If more than 50 outer iterations are required, terminate the iteration, accept the current solution, and write a caution message.
13. Compute thermal fissions and the slowing-down density from equation (16) and (27).
14. If the last rebalancing factor differs from unity by more than 10^{-5} , write a caution message.

At the present time resonance capture in the thermal range is not included. If a suitable expression for the effective slowing-down density is developed, resonance capture can be easily included.

JOSHUA DPSTM-500	Title Module FIST	Vol. IV	Page E.2.16
		Rev.	Date 6/70

2.5 Fast Flux Calculation (FFLUX)

For an active lattice, an outer iteration must be used to settle the fast fission distribution. Since most of the fission occurs at thermal and epithermal energies, few iterations will be required. The calculation proceeds as follows:

1. Start fission distribution iteration.
2. Calculate a fission source distribution using subroutine FISØR.
3. Calculate a fast flux by group and sub-region using subroutine EFLUX.
4. Compare the fast fission production from this outer iteration with those from the preceding iteration. If the change in the fast production divided by the total production in all ranges for any isotope or sub-region is greater than 10^{-4} , do another iteration starting from step 2. If more than 10 outer iterations are required, accept the current distribution and write a caution message.

If the lattice is dead (no fissions), a spatially uniform fission spectrum at a temperature of 1.273 Mev is established and a fast flux calculated using subroutine EFLUX.

2.6 Fission Source Calculation (FISØR)

Fission production $F'_k(m,r)$ computed from an equation like (22) is stored by isotope (k), sub-region (m), and energy range (r).

$$F'_k(m,r) = \sum_{i \in r} v_k(i) \left\{ \Sigma_k^f(m,i) \phi(m,i) \Delta E(i) V(m) + R_k^{*f}(m,i) Q(c,i) \right\} \quad (30)$$

Define the following terms:

- A_1 - Total absorptions in energy range 1.
- A_{23} - Total absorptions in energy ranges 2 and 3.
- L_1 - Total leakage in energy range 1.
- L_{23} - Total leakage in energy ranges 2 and 3.
- P_1 - Total fission production in energy range 1.
- P_{23} - Total fission production in energy ranges 2 and 3.

At the start of the first outer iteration on the fast fission distribution, the total reactions for energy ranges 2 and 3 are known, but nothing is known about energy range 1. An eigenvalue, k_{eff} , is estimated from

$$k_{eff} = P_{23} / (A_{23} + L_{23}) \quad (31)$$

and a fission source computed from

$$H_1(m,j) = \frac{1}{k_{eff}} \sum_k x_k(j) \sum_r F'_k(m,r) / V(m) \quad (32)$$

with r limits of 2 and 3. When at least one outer iteration has been done, there is an estimate of the total fast reactions. The eigenvalue is then estimated from

$$k_{eff} = (P_1 + P_{23}) / (A_1 + A_{23} + L_1 + L_{23}) \quad (33)$$

and a fission source computed from (32) with r limits of 1 and 3.

2.7 Spatial Flux Calculation (GRITER)

Subroutine GRITER solves the one-group transport equation by iteration given a source into the group, the absorption, removal, and in-group scattering cross sections, and a starting guess for the flux and currents. The steps are as follows:

1. Obtain the transport matrix (TMAT blocks) for this energy group.
2. If no starting guess is given, establish a spatially uniform flux and zero current guess.
3. Scale the starting guess so that total absorption plus leakage plus removal equals the total source.
4. Start the spatial iteration.
5. Compute the flux in each sub-region using equation (1) and over-relaxing by a factor of 1.2. The inward currents and fluxes used in the RHS of (1) are those from the preceding iteration.
6. Compute the outward current using equation (2). The inward currents used in the RHS of (2) are those from the preceding iteration. The fluxes used in the RHS of (2) are those from step 5.
7. Compute a new set of inward currents using equation (5).
8. Scale the fluxes and currents so that total absorption plus leakage plus removal equals the total source.
9. Compare the fluxes from this iteration with those from the preceding iteration. If the fractional changes in all sub-regions is less than 5×10^{-5} , the solution has converged.
10. If the flux has not converged, predict a new set of fluxes and currents by quadratic extrapolation of the last three iterations.
11. The maximum number of iterations allowed is specified by the program calling GRITER. If this maximum is exceeded, accept the current solution.

2.8 Group Constants Calculation (GRPCØN)

The subroutine GRPCØN calculates macroscopic cross sections and sources for one energy group (J) and all sub-regions. The cross sections calculated are absorption, fission, neutrons/fission, removal, and in-group (see equation (14)). Microscopic cross sections are supplied by subroutine ISØDAT for each isotope. GRPCØN forms macroscopic cross sections from these and the concentration data in the CØMP block.

Source terms are calculated using equation (13) and a range of initial groups i supplied as arguments to GRPCØN. Subroutine ISØDAT supplies microscopic scattering matrix elements and GRPCØN forms macroscopic matrix elements from these and the data in the CØMP block.

GRPCØN controls the storage of the XSEC and SMAT blocks. If XSEC and SMAT blocks are not in the DATA array for group J, all existing XSEC and SMAT blocks are deleted and the appropriate ones read into the DATA array from the LIDS.

JOSHUA DPSTM-500	Title Module FIST	Vol. IV	Page E.2.20
		Rev.	Date 6/70

2.9 Isotope Data Retrieval (ISØDAT)

Subroutine ISØDAT fetches microscopic cross sections and scattering matrices for one final energy group and one isotope from the XSEC and SMAT blocks. If only the down-scattering terms of the thermal scattering matrix are given in SMAT block, ISØDAT computes the up-scattering terms from the detailed balance equation.

JOSHUA DPSTM-500	Title Module FIST	Vol. IV	Page E.2.21
		Rev.	Date 6/70

2.10 Resonance Rates (RESSR)

Subroutine RESSR fetches absorption and fission rates from the RESD block by isotope and sub-region for one energy group. These rates are corrected using equations (17 - 19) and the flux in the preceding energy group.

3. Coding Details

3.1 Module Structure

The FIST module consists of a main program, subroutines, and COMMON areas. The hierarchy is as follows:

FIST (main program)

- ITSET (initialization)
- FFLUX (fast flux calculation)
- EFLUX (epithermal flux calculation)
- TFLUX (thermal flux calculation)
- CRF (compute region flux)
- RESWGT (correct resonance rates)
 - GRPCØN (get group constants)
 - FISØR (calculate fission source)
 - GRITER (one-group spatial iteration)
 - ISØDAT (get isotope data)
 - RESSR (get resonance rates)
 - MANAGE (data manager; see manuscript B.1)

- DATA (COMMON area)
- SPECS (COMMON area)
- GRØUPS (COMMON area)
- TEMP (COMMON area)
- ITSTAT (COMMON area)
- SCRACH (COMMON area)

The logic for most of the subroutines is adequately covered in the previous sections. A few additional comments on GRITER and ISØDAT are provided in the next two sections. The remaining sections define the symbols used in the COMMON areas.

3.2 Subroutine GRITER

Subroutine GRITER is called with the statement

CALL GRITER (FLUX,JIN,JØUT,ITMAX,ITC,TAB,TLK,TFLX,I)

- FLUX(NSR) - The flux in each sub-region NSR. (On entry this array contains the starting guess).
- JIN(NFC) - The inward current through each face NFC. (On entry this array contains the starting guess).
- JØUT(NFC) - The outward current through each face NFC. (On entry this array contains the starting guess).
- ITMAX - Maximum number of spatial iterations allowed.
- ITC - Actual number of spatial iterations required. If ITC=ITMAX, the solution had not converged.
- TAB - Total absorptions in the supercell.
- TLK - Total leakage from the supercell.
- TFLX - Total (volume integrated) flux in the supercell.
- I - Energy group number being considered.

The logic used in subroutine GRITER is given in section 2.7.

JOSHUA DPSTM-500	Title Module FIST	Vol. IV	Page E.2.24
		Rev.	Date 6/70

3.3 Subroutine ISØDAT

Subroutine ISØDAT is called with the statement

CALL ISØDAT(NIS,J,IL,IH,S,XS,KXØRG,KSØRG)

- NIS - isotope number
- J - final group number
- IL,IH - low and high initial group numbers
- S(I) - scattering matrix column, $\sigma^S(I \rightarrow J) * DE(I)$,
I=IL,IH.
- XS(N) - cross section list
 - N=1, Total
 - =2, Absorption
 - =3, Removal (excluding absorption)
 - =4, Transport (including absorption)
 - =5, Fission
 - =6, ν (neutrons/fission)
- KXØRG - origin (first word minus one) of the XSEC
block in the DATA array for the current energy
band
- KSØRG - origin (first word minus one) of the SMAT
block in the DATA array for the current energy
band

Cross sections from the MULTIGRP data are stored in the XSEC block by data type index, NDT.

<u>NDT</u>	<u>Type</u>
1	Total, $\sigma^t(J)$
2	Capture = $(n,\nu)+(n,p)+(n,d)+etc., \sigma^c(J)$
3	Fission, $\sigma^f(J)$
4	$\nu(J)$
5	Transport (excluding absorption, including n2n), $\sigma^{tr}(J)$
6	$(n,2n), \sigma^{n2n}(J)$

The scattering matrices in the SMAT block are defined by

$$\sigma^S(I \rightarrow J) = \sigma^{el}(I \rightarrow J) + \sigma^{in}(I \rightarrow J) + 2\sigma^{n2n}(I \rightarrow J)$$

where "el" and "in" denote elastic and inelastic. The cross section $\sigma^S(J)$ associated with the matrix is also stored in the SMAT block. These data are moved into the output arrays as follows:

$$S(I) = \sigma^S(I \rightarrow J) \Delta E(I) \quad I \neq J$$

$$S(J) = \sigma^S(J \rightarrow J) \Delta E(J) + \sigma^{tr}(J) - \sigma^S(J)$$

$$XS(1) = \sigma^t(J)$$

$$XS(2) = \sigma^c(J) + \sigma^f(J) - \sigma^{n2n}$$

$$XS(3) = \sigma^S(J) - \sigma^S(J \rightarrow J) \Delta E(J)$$

$$XS(4) = [\sigma^{tr}(J) - 2\sigma^{n2n}(J)] + [\sigma^c(J) + \sigma^f(J) + \sigma^{n2n}(J)]$$

$$XS(5) = \sigma^f(J)$$

$$XS(6) = v(J)$$

JOSHUA DPSTM-500	Title Module FIST	Vol. IV	Page E.2.26
		Rev.	Date 6/70

3.4 CØMMØN/DATA/

The area CØMMØN/DATA/ contains the array DATA(K) into which are packed most of the LIDS blocks. The blocks used here are CØMP, GEØM, ZØNE, RESD, CHI, FLUX, JIN, JØUT, QCEL, XSEC, SMAT, and TMAT. The last three types are rolled in and out of the DATA array as data for each energy band is needed. Refer to manuscript B.2 for the structure of these LIDS blocks.

3.5 COMMON/SPECS/

Specification information is contained in this area. The following data are included:

TDSN	-	Temporary data set name = 'TEMP'	(R*8)
DLAT	-	Lattice name	(R*8)
VLAT	-	Lattice version name	(R*8)
NCASE	-	Case number	
IDØ(NZN)	-	Test for the occurrence of identical transport matrices (TMAT) in several zones. If IDØ(NZN) = 0, a TMAT block exists for zone NZN. If IDØ(NZN) > 0, a separate TMAT block does not exist and the block for zone NZNP=IDØ(NZN) is to be used.	

JOSHUA DPSTM-500	Title Module FIST	Vol.	Page
		IV	E.2.28
		Rev.	Date
			6/70

3.6 COMMON/GRUPS/

Group structure information taken from the GRPS record on the LIDS is stored in this area.

- NEG - Number of energy groups.
- JFTG - Number of the first thermal group.
- JCHI - Number of the last group where some fission spectrum is non-zero.
- IDBF - Detailed balance flag (0 = full scattering matrices given, 1 = half-matrices given).
- E(I) - Upper energy limit (ev) of group I.
- EBAR(I) - Characteristic energy (ev) of group I.
- DE(I) - Width (ev) of group I.
- NEBX - Number of energy bands.
- IEBH(NEB) - Largest group number contained in energy band NEB.

3.7 CØMMØN/TEMP/

The CØMMØN/TEMP/ area is the main area used to pass data between the subroutines of FIST. The following general data are included:

NSRX	-	Number of sub-regions in supercell.
NFCX	-	Number of faces in supercell.
NCLX	-	Number of cells in supercell.
NFX	-	Current number of fission production entries.
NFXX	-	Maximum value of NFX allowed.
ITØUT	-	ITØUT=1 during the first pass through the epithermal and thermal ranges. ITØUT is used along with LØLDF to select a starting flux guess.
TABS	-	Current total absorptions in the supercell.
TLEAK	-	Current total leakage from the supercell.
EFFK	-	Current value of k_{eff} .
LØLDF	-	Old flux test. If LØLDF $\neq$ 0, a starting flux was read in from a previous problem.
VSR(NSR)	-	Volume of sub-region NSR.
AFC(NFC)	-	Area of face NFC.
KMTT(NSR)	-	Absolute pointer to the material in the CØMP block used in sub-region NSR.
NCLT(NSR)	-	Cell number in which sub-region NSR occurs.
XING(NSR)	-	The in-group scattering cross section for sub-region NSR and the current energy group.
HP(NSR)	-	The total source into sub-region NSR and the current energy group.
XABS(NSR)	-	The absorption cross section for sub-region NSR and the current energy group.
NIST(NF)	-	Isotope number for fission production term NF.
NSRT(NF)	-	Sub-region number for fission production term NF.

- PRØD(NF,M) - Total fission production for term NF from energy range M. M = 1 is fast, 2 is epithermal, 3 is thermal, and 4 is temporary storage of fast production during the fast fission iteration. Each term indexed by NF is for isotope (NIST(NF) in sub-region NSRT(NF).
- VNT(NF) - ν (neutrons/fission) for production term NF and the current energy group.
- XFT(NF) - Smooth fission cross section for production term NF and the current energy group.
- RAT(NF) - Resonance absorption rate for production term NF and the current energy group.
- RFT(NF) - Resonance fission rate for production term NF and the current energy group.
- SØURCE(NSR,JP) - Source of neutrons into sub-region NSR and energy group index JP. When the fast and epithermal ranges are being calculated, SØURCE contains the normalized fission distribution and JP=J, the group number. When the thermal range is being calculated, SØURCE contains the epithermal source to the thermal range and JP=J-JFTG+1.
- TLPSR(NSR) - Total number of lattice positions for the cell in which sub-region NSR occurs.
- TLPFC(NFC) - Total number of lattice positions for the cell in which face NFC occurs.
- TLPCL(NCL) - Total number of lattice positions for cell NCL.
- TPRØD(M) - Total fission production for the fast (M=1), epithermal (M=2), and thermal (M=3) energy ranges.

3.8 COMMON/ITSTAT/

Selected iteration statistics are stored in this area in the array ITST(L,M,ITOUT).

L=1, Number of outer iterations performed.

=2, Largest number of inner iterations performed by any entry to subroutine GRITER.

=3, Total number of inner iterations performed by subroutine CRITER.

=4, Number of entries to subroutine GRITER.

M=1, Fast energy range.

=2, Epithermal energy range.

=3, Thermal energy range.

ITOUT=1, First pass through the epithermal and thermal ranges.

=2, Second pass through the fast, epithermal, and thermal ranges.

Normally, this record is created by the RAHAB control module. These items are used to establish the names of the input and output records.

4.2 Output

The following records are written to the LIDS by FIST:

('TEMP'. 'LIDS'.DLAT.VLAT → V)

V.'FLUX'.NCASE.0.0	-	flux by sub-region and group
V.'FLUX'.NCASE.1.0	-	flux by region and group
V.'JIN' .NCASE.0.0	-	ingoing currents by face and group
V.'JØUT'.NCASE.0.0	-	outgoing currents by face and group
V.'QCEL'.NCASE.0.0	-	slowing-down density by cell and group
V.'RESD'.NCASE.1.0	-	resonance rates corrected for flux fine-structure.

See manuscript B.2 for a description of these records.

4.3 Core Size and Timings

The size of the various subroutines and COMMØN areas are given in Table I. Timings are also given in Table I for a calculation with 84 groups, 2 zones, and 20 sub-regions.

4.4 Error Stops

All stops are handled by the JOSHUA operating system and are summarized in Table II.

TABLE I

SIZE AND TIMING FOR FIST SUBPROGRAMS
(Size in bytes, time in CPU seconds.)

Subprogram	Size	Total Time	Average Time	Entries
MAIN	3,754	.11648	.11648	1
GRITER	4,046	63.36512	.06286	1,008
FFLUX	1,464	-	-	1
EFLUX	2,522	.59904	.09984	6
TFLUX	3,912	1.49760	.74880	2
ISØDAT	2,152	14.80960	.00188	7,896
FISØR	1,242	.03328	.00832	4
ITSET	2,004	-	-	1
GRPCØN	2,444	36.72448	.03256	1,128
CRF	1,148	.11648	.11648	1
RESWGT	1,414	.03328	.03328	1
RESSR	1,842	.38272	.00228	168
MANAGE	12,332*	16.08104**	.00071	22,614
IHCSERF	456			
IHCSEXP	432			
IHCSSQRT	329			
/DATA/	100,000			
/TEMP/	24,092			
/GRØUPS/	1,624			
/SPECS/	68			
/ITSTAT/	96			
/SCRACH/	1,600			
/DATCAT/	2,440			
JØS	80,451***			
TOTALS	251,904	133.75912		32,831

* Includes all routines of data manager.

** Includes time for JOSHUA Operating System and output routines.

*** Includes JOSHUA Operating System and buffers.

Table II

Error Stop Messages

Message/Explanation

FIST.MAIN, GRPS RECORD MISSING

The group structure record is not in the LIDS.

FIST.MAIN, CONTROL RECORD MISSING

The record INPUT.FIST is missing.

FIST.MAIN, POINTER MISSING

One of the blocks COMP, GEOM, ZONE, RESD, or CHI is not in the LIDS.

FIST.GRITER, POINTER MISSING

The ZONE block is not in the DATA array.

FIST.GRITER, TMAT BLOCK MISSING

The TMAT block is neither in the DATA array nor in the LIDS.

FIST.GRITER, TMAT ERROR

There is no data for group I in the TMAT block.

FIST.EFLUX, POINTER MISSING

One of the blocks FLUX, JIN, JOUT, QCEL is not in the DATA array.

FIST.TFLUX, POINTER MISSING.

One of the blocks FLUX, JIN, JOUT, QCEL is not in the DATA array.

FIST.GRPCON, POINTER MISSING

One of the blocks COMP or FLUX is not in the DATA array.

FIST.GRPCON, XSEC OR SMAT BLOCKS MISSING

One of the blocks XSEC or SMAT is not in the LIDS.

JOSHUA DPSTM-500	Title Module FIST	Vol. IV	Page E.2.36
		Rev.	Date 6/70

Table II

Error Stop Messages

(Cont.)

Message/Explanation

FIST.CRPCON, TØØ MANY ISØTØPE/SUBREGION PAIRS

The number of fissionable isotope/subregion pairs exceeds 200.

FIST.ISØDAT, CØMP BLOCK MISSING

The CØMP block is not in the DATA array.

FIST.ITSET, PØINTER MISSING

One of the blocks ZØNE, GEØM, CØMP is not in the DATA array.

FIST.ITSET, GEØM ERRØR

The cell number from the ZØNE block is not in the GEØM block.

FIST.ITSET, TØØ MANY SUBREGIONS

The number of subregions exceeds 100.

FIST.ITSET, TØØ MANY FACES

The number of faces exceeds 100.

FIST.ITSET, CHI TØØ LARGE

The highest group number for which CHI data is given is larger than 30.

FIST.ITSET, TØØ MANY THERMAL GRØUPS

The number of thermal groups exceeds 30.

FIST.CRF, PØINTER MISSING

Either the FLUX or CØMP block is not in the DATA array.

FIST.CRF, DATA ARRAY ØVERFLØW

Too much data to pack into the DATA array.

FIST.RESWGT, PØINTER MISSING

Either FLUX, RESD, or GEØM is not in the DATA array.

DISTRIBUTION

1. F. P. Baranowski	42. J. E. Beach
2. J. Lewellyn	43. R. M. Wallace
3. I. A. Hobbs	44. J. A. Porter
4. P. J. Hagelston	45. F. R. Field
5. T. B. Niland	46. D. H. Knoebel
6. R. L. Smith	47. I. M. Macafee
7. J. R. Jakes	48. R. C. Thornberry
8. C. W. Farmer, Jr.	49. J. R. Taylor
9. C. A. McCharen	50. Jerry Robertson
10. J. W. Croach	51. J. R. Hilley
11. F. E. Kruesi	52. R. M. Satterfield
12. B. C. Kline	53. M. B. Stroud
13. R. F. Gambill, Jr.	54. R. A. Cooper
14. R. G. Croom	55. W. H. Brotherton, Jr.
15. W. B. Daspit	56. J. P. Church
16. W. M. Olliff	57. F. D. Knight
17. O. A. Towler	58. F. D. King
18. E. O. Kiger	59. W. V. Baxter
19. D. A. Ward	60. R. J. Pryor
20. P. S. Tennant	61. M. R. Buckner
21. J. N. Wilson	62. C. K. Paulson
22. A. H. Peters	63. B. Toppel - ANL
23. L. H. Meyer	64. Richard Ehrlich - KAPL
24. J. E. Suich	65. D. Vondy - ORNL
25. H. C. Honeck	66. C. W. Craven, Jr. - ORNL
26. L. M. Arnett	67. R. L. Howell - Oak Ridge
27. W. R. Hartshorn	68. P. Michael - BNL
28. J. C. Jensen	69. C. W. Lindenmeier - BMI
29. R. S. Wood	70. G. H. Best - LASL
30. C. E. Bailey	71. R. A. Dannels - WAPD
31. D. R. Finch	72. D. Roy - B&W
32. W. H. Hightower	73. P. G. Aline - GE
33. J. H. Maddox	74. B. Hutchins - GE
34. J. E. Bearden	75. Alleen Thompson - GE
35. J. W. Stewart	76. H. Alter - AI
36. B. P. Compton	77. R. F. DeAngelis - U. N. Corp.
37. W. E. Graves	78. Martin Becker - RPD
38. D. A. Sharp	79-80. D. S. Craig - AECL
39. F. J. McCrosson	81-99. TIS File
40. P. L. Ames	100. Vital Records
41. D. E. Hostetler	