NUCLEAR ENGINEERING READING ROOM - M.I.T.

MITNE-216 C.2

AN IMPROVED LONG RANGE FUEL MANAGEMENT PROGRAM

Ъy

C. L. Beard

May, 1978

Massachusetts Institute of Technology Department of Nuclear Engineering Cambridge, Massachusetts

Support provided by: Westinghouse Electric Corporation Yankee Atomic Electric Company

ABSTRACT

A new procedure has been developed for rapid and accurate long range fuel management studies. A computer program, FLAC, was written employing this new procedure. The program has been verified and qualified for use on large, commercial, nuclear reactors. Sensitivity studies have been performed to determine the best methods for use of the program.

The procedure is derived from the FLARE model of nodal coupling. The nodal equations are combined and averaged over regions to generate coupled equations on a region-wise basis. This set of coupled equations and its corresponding eigenvalue is solved to determine the relative region fission sharings and the core average k_{off} .

The FLAC program automatically performs the cycle depletions and has the capability of determining (1) cycle burnup, (2) end of cycle keff or (3) the required feed enrichment. Up to 20 sequential cycles or equilibrium cycle calculations can be performed. A significant number of options have been included in the program for easy utilization without requiring detailed knowledge of future cores. For example, the program can generate data pertinent to either a ring interior or a checkerboard interior assembly arrangement. The ability to use basic loading pattern data is a significant improvement over the currently used weighted k procedures. And the ability to automatically generate the loading pattern data provides for an ease of usage unattainable with multi-dimensional calculations. Thus the FLAC program allows one to do detailed fuel management studies with a simplicity of input preparation.

ACKNOWLEDGEMENTS

This report has been submitted in partial fulfillment of the requirements for the Master of Science degree at MIT. Professor David D. Lanning and Professor Michael J. Driscoll served as thesis advisors and their time, support and assistance is appreciated.

I gratefully acknowledge the support given me by the Westinghouse Electric Corporation. Without its support I would have never had this opportunity. I would also like to acknowledge the financial and technical support given by the Yankee Atomic Electric Company for the completion of this research. Special thanks are due to Ed Pilat, whose ideas spurred this project. Of course, this project could never have been completed without the support and help of the MIT Staff and students. Special thanks are also in line for Rachel Morton whose invaluable aid greatly assisted in my familiarization with the MIT Computer system. Last, but not least, is my gratitude for my wife, Donna, and my children, Sheryl and Patrick, for their willingness to temporarily relocate our home and especially for always providing me with the needed diversion to get away from it all.

TABLE OF CONTENTS

	Page
ABSTRACT	2
ACKNOWLEDGEMENTS	
TABLE OF CONTENTS	
LIST OF TABLES	
LIST OF FIGURES	7
CHAPTER 1 INTRODUCTION	8
1.1 Background	9
1.2 Summary	10
CHAPTER 2 ANALYTICAL METHODS	13
2.1 Neutronics Model	13
2.2 Matrix Solution	17
2.3 Generation of Tabular Data	18
2.4 Generation of Shuffle	22
2.5 Generation of Edge Data	22
2.6 Interpolation Scheme	2 6
2.7 Burnup Calculation	2 8
2.8 Discharge Burnups	3 0
2.9 Burnable Poison Treatment	30
2.10 Cycle Search Procedure	31
2.11 Convergence Criteria	32
CHAPTER 3 VERIFICATION AND QUALIFICATION 34	
3.1 Verification	34
3.1.1 Matrix Verification	35
3.1.2 Interpolation Verification	38

.

3.2 Qualifica	ation	40
3.3 Sensitivit	ty Analysis	48
3.3.1 Ser	nsitivity to W and Alpha	48
3.3.2 Ser	nsitivity to Edge Data	49
3.3.3 Ser	nsitivity to Boron Search	59
CHAPTER 4 CONCLUSIO	ONS AND RECOMMENDATIONS	64
4.1 Recommende	ed Method	64
4.2 Method Acc	uracy	66
4.3 Conclusion	15	66
4.4 Future Wor	•k	67
CHAPTER 5 REFERENCE	lS	70
APPENDIX A FLAC USE	R'S MANUAL	71
A.1 Input Prep	paration	71
A.1.1 Inp	out Description	71
A.1.2 Sam	ple Input	93
A.2 Program Ou	itput	9 6
A.2.1 Out	put Description	96
A.2.2 Sam	aple Output	99
A.3 Programmer	's Information	123
A.3.1 Pro	gram Routines	123
A.3.2 Pro	gram Variables	135
APPENDIX B PROGRAM	LISTING	132

LIST OF TABLES

Number	Title	Page
1-1	Nomenclature	12
2-1	Convergence Criteria	33
3-1	Matrix Solution Test Problem	37
3-2	Data for k_{∞} Interpolation	39
3-3	ZION 64 Feed Assembly Cycle Study	42
3-4	ZION 48 Feed Assembly Cycle Study	43
3-5	ZION Variation from 64 Feed Assemblies	44
3-6	ZION Variation from 48 Feed Assemblies	45
3-7	Burnup Sharing Comparison	47
3-8	Loading Pattern Sensitivities	58
3-9	Equilibrium Feed Enrichment	60
3-10	Sensitivity to Boron Search	62
A-1	Input Parameters	72
A-2	Program Variables	127

:

FIGURES

Number	Title	
2-1	Axial Leakage Factor	
3-1	Sensitivity Study, k_{eff} versus ω and α	50
3-2	Sensitivity Study, k_{eff} versus ω and α	51
3-3	Sensitivity Study, Region 1 Power versus ω and α	52
3-4	Sensitivity Study, Region 1 Power versus ω and α	53
3-5	Sensitivity Study, Region 2 Power versus ω and α	54
3-6	Sensitivity Study, Region 2 Power versus ω and α	55
3-7	Sensitivity Study, Region 3 Power versus ω and α	56
3-8	Sensitivity Study, Region 3 Power versus ω and α	57
3-9	Equilibrium Feed Enrichment	61
A-1	Program Flowchart	124

3

۱.

CHAPTER I

INTRODUCTION

Nuclear fuel management is one of the major responsibilities of a utility with nuclear power plants. The objective is to obtain the maximum utilization of the fuel while meeting scheduling and safety constraints. There are currently a number of computational tools that are used in this area, ranging from the trivial (5,7,8,12) to the complicated (3,11). However, the trivial require significant normalization and the complicated are very costly and time-consuming. Therefore, a computer program has been developed embodying the benefits of both the trivial and the complicated tools.

The model as developed makes use of the FLARE (4) equations on a regionwise basis. Thus it keeps the regionwise data of the trivial codes but solves for the k_{eff} and power distribution like the complicated codes.

A computer code, FLAC, has been written by using the model as described in this report. The program incorporates many options and automated input preparation to enable it to be easily used. It has been thoroughly tested, verified and qualified. Its usefulness has been demonstrated with numerous studies which can be done quickly and easily.

The main thrust of this development has been the modeling of pressurized water reactor cores and all the test cases on the computer program have been with PWR's. However, it is felt that these methods could be extended to boiling water reactors, but it is left for future work.

I.1 Background

Currently there are many ways for doing long range fuel management varying from the trivial to the very difficult and complex. The FLAC program has been developed to lie between the simple methods and the difficult and time consuming methods.

The simple methods (5,7,8,12) are based on variations of the general formula:

$$\mathbf{k}_{av} = \mathbf{f}^{-1} \left\{ \left[\sum_{i=1}^{N} \mathbf{w}_{i} \mathbf{f}(\mathbf{k}_{i}) \right] / \sum_{i=1}^{N} \mathbf{w}_{i} \right\}$$
(1.1)

where

- k is the core average k,
 - N is the number of regions
- w, is a weight for region i
- f(.) is a general function which has an inverse $f^{-1}(.)$

The simple linear reactivity model assumes that f(k) = k and that $w_i = n_i$, the number of assemblies in region i. Other models use a linear reactivity model but use a weighting function equal to the region power times the number of assemblies.

The big drawback to all of these simple methods is the need to know the region-wise power sharings to perform the calculation. Even if the weight functions do not include the power, it is still required to determine the end of cycle region-wise burnups. Thus significant engineering judgement is required or an equivalent multi-dimensional calculation.

This greatly increases the difficulty of use, especially for novel problems. Procedures of fitting the power sharings to various parameters have been tried, but these at best yield only a first approximation to the actual values. There is too much coupling in the core to adequately determine the power sharings with a simple fit. Also, the simple calculations can not properly factor in radial core leakage.

There are a variety of complicated methods for long range fuel management (3,11), but they all require dimensional depletion calculations. To do these depletions, loading patterns are required for each cycle. Therefore, in addition to the large computer expenditure, a considerable amount of man-time is required to setup these models and to find suitable loading patterns. These models usually give accurate results, but unfortunately reality seldom follows the desired path. Due to some circumstance such as plant maintenance and/or operating requirements, the actual fuel management scheme can not be followed as planned. Then all the fuel management work is in error because a base point has now been changed. Therefore, although it is possible to accurately predict core behavior for a number of cycles, it is very costly and quite likely to be invalidated by uncontrollable events. Therefore, there is little incentive to perform these detailed calculations.

I.2 Summary

The FLAC computer program has been written to provide a long range fuel management tool which generates results of the desired accuracy with a minimum of effort and cost. This report describes the model, the verification, the method of use and provides information about the computer program.

Chapter 2 describes the basic model used in FLAC as well as the many options available in the program. These options include automated generation of the fuel shuffling data from cycle to cycle and the edge correspondence between regions for either a ring or a checkerboard loading pattern. Also discussed is the interpolation scheme, the matrix solution procedure and the various iterative search procedures employed by the program.

The third chapter discusses the verification and qualification of the computer code and the sensitivity to the important input parameters. The verification demonstrates that the code has been properly programmed as specified in the model. The qualification shows that the program is applicable for long range fuel management studies for large PWR cores. The results of sensitivity analysis can be used to determine the importance of the various input parameters as an aid in deciding how much effort should be put into input generation.

Chapter 4 presents the recommended method of use of the FLAC computer program based on the previous qualification and sensitivity analyses. The method is described and the anticipated accuracy is defined. The conclusions of the study are also given along with many ideas for future work.

There are two appendices to this report, the first is a user's manual for the FLAC program and the second is a listing of the program. The user's manual includes a detailed input description and a discussion of the output edits. Sample input and output listings are given as an aid to understanding. Also included is a section giving programmer's information which includes descriptions of the various variables and the subroutines in the program.

As an aid to understanding this report a table of nomenclature has been included as Table 1-1.

TABLE 1-1

NOMENCLATURE

Symbol	Definition
α	Average albedo for an assembly
BOC	Beginning of cycle
BP	Burnable poison
Bu	Burnup
BWR	Boiling water reactor
EFPH	Effective full power hours
EOC	End of cycle
ε	Initial enrichment
$F_{\Delta H}$	Hot channel peaking factor
GWD/MTM	Giga-wall days per metric ton metal
К	Energy release per fission
^k eff	Core effective k
k _∞	Infinite core k
λ	Fundamental eigenvalue
MWD/MTM	Megawatt days per metric ton metal
n	Number of edges between regions
N	Number of regions or number of assemblies
ν	Neutrons produced per fission
ω	Over-relaxation parameter
PWR	Pressurized water reactor
S	Fission Source
Σ_{f}	Macroscopic fission cross section
W	The probability that a neutron produced in one assembly will be absorbed
w/o	Weight per cent, generally w/o of U-235 in the uranium fuel

CHAPTER 2

ANALYTICAL METHODS

FLAC is a computer program which is based on very simple, fundamental assumptions. As such, it is very easy to comprehend the analytical model underlying the program. However, in order to make FLAC a very useful program, many different conveniences were built into the code. These include various interpolation schemes, search procedures and input generation. The basic models and assumptions underlying all of these aspects are presented in this section.

2.1 Neutronics Model

Starting with the basic FLARE (4) equation:

$$S_{j} = \frac{\frac{k}{\Delta} \sum S W}{1 - \frac{k}{\lambda} M_{jj}}$$
(2.1)

where S_j = rate of production of fission energy neutrons at node; where each node is one fuel assembly W_m = probability that a neutron born at node m will ultimately

- be absorbed at node j
- $k_{\substack{\infty j}}$ = $k_{\substack{\infty}}$ at node j including xenon and axial affects
 - λ = fundamental eigenvalue

Also, we know that:

$$W_{jj} = 1 - \sum_{m jm} W_{jm} - \sum_{m'} (1 - \alpha_{m'}) W_{jm'}$$
 (2.2)

where:

m' are the edges on the exterior

m are the edges on the interior

and α_m , is the albedo on an exterior edge.

If W_{mj} is independent of j, then $W_{mj} = W_m$. Also, m + m' = 4 for every node. Therefore, equation 2.2 can be rewritten as:

$$W_{jj} = 1 - 4W_{j} + \sum_{m} \alpha_{m}, W_{j}$$
 (2.3)

Combining equations 2.1 and 2.3 gives:

$$S_{j} = \frac{\frac{k_{\infty j}}{\lambda} \sum_{m} S_{m} W_{m}}{1 - \frac{k_{\infty j}}{\lambda} [(1 - 4W_{j}) + \sum_{m} \alpha_{m}, W_{j}]}$$
(2.4)

which can be rewritten as:

$$\left[\frac{\lambda}{k_{\infty j}} - (1 - 4W_j) + \sum_{m} \alpha_m, W_j\right] S_j = \sum_{m} S_m W_m$$
(2.5)

Assume that $k_{\infty j}$ and W_j are the same for each node in region I. This assumption implies that all the assemblies in a region are identical. That is they have the same enrichment, burnup and burnable poison content. Then summing over all the nodes in region I gives:

$$N_{I}S_{I}\left[\frac{\lambda}{k_{\infty I}} - (1 - 4W_{I})\right] + \alpha_{I}S_{I}W_{I}n_{I\alpha} = \sum_{J=1}^{N} S_{J}W_{J}n_{IJ}$$
(2.6)

for I = 1, 2, ..., N

where:

 N_{I} is the number of assemblies in region I S_{I} is the average fission source in region I W_{I} is the average absorption probability in region I α_{I} is the average albedo in region I n_{IJ} is the number of edges of region I adjacent to region J $n_{I\alpha}$ is the number of assembly edges in region I on the exterior and N is the number of regions.

Note that $n_{IJ} = n_{JI}$ and that $4N_I = \sum_{J=1}^{N} n_{IJ} + n_{I\alpha}$. Equation 2.6 can be rewritten as:

$$\{[(1 - 4W_{I}) - 4\alpha_{I}f_{I\alpha}W_{I}]k_{\infty I} - \lambda\}s_{I} + \sum_{J=1}^{N} (4W_{J}f_{IJ}k_{\infty I})s_{J} = 0$$

for
$$I = 1, 2, ..., N$$

where

 $f_{I\alpha} = \frac{n_{I\alpha}}{4N_{I}} =$ fraction of edges on the exterior

 $f_{IJ} = \frac{n_{IJ}}{4N_{I}} =$ fraction of edges in region I adjacent to region J

$$f_{I\alpha} + \sum_{J=1}^{N} f_{IJ} = 1.0$$
 (2.8)

If one defines

$$\beta_{I} = [(1 - 4W_{I}) - 4\alpha_{I}f_{I\alpha}W_{I}]k_{\infty I}$$

$$(2.9)$$

and $\gamma_{IJ} = 4W_J f_{IJ} k_{\infty I}$ (2.10)

Then Equation 2.7 can be simply written as:

$$(\beta_{I} - \lambda)S_{I} + \sum_{J=1}^{N} \gamma_{IJ}S_{J} = 0$$
 (2.11)
for I = 1, 2, ..., N

. . .

which is a standard eigenvalue problem.

The matrix formulation for equation 2.11 is:

Solving for the largest eigenvalue will give the approximate core eigenvalue. Its eigenvector will give the approximate assembly average source terms.

The power sharing can then be found from

$$P_{I} = \left(\frac{\kappa \Sigma_{f}}{\nu \Sigma_{f}}\right) I^{S} I = \left(\frac{\kappa}{\nu}\right) I^{S} I$$
(2.13)

where $\begin{pmatrix} \kappa \\ \nu \end{pmatrix}_{I}$ is the number of neutrons per fission divided by the average energy production per fission for region I.

The power sharings are then normalized so that the volume weighted average is unity.

2.2 Matrix Solution

The matrix generated is a very small (on the order of 3 x 3 to 7 x 7) matrix of the same form as in the FLARE program. We know that it is a positive definite matrix within the context of its representation of a reactor core and its representative parameters. The largest eigenvalue will be distinct and positive and its eigenvector positive. Therefore, a straight power iteration with an initial unity estimate for the eigenvector will generate the desired eigenvalue and eigenvector.

As an aid to convergence, an over-relaxation procedure is used of the form:

$$s^{i+1} = s^{i} + \omega([A]s^{i} - s^{i})$$
(2.14)

where ω is the over-relaxation parameter. The default value built into the code for ω is 1.80.

The iteration proceeds until convergence which is measured by two parameters:

$$\Delta_1 = \max \left(\left| S_j^{i+1} - S_j^i \right| \right)$$

and

$$\Delta_2 = |\lambda^{\mathbf{i+1}} - \lambda^{\mathbf{i}}|.$$

Convergence is assumed if $\Delta_1 \leq 1 \ge 10^{-3}$ and $\Delta_2 \leq 1 \ge 10^{-5}$. With the small matrices encountered, a maximum of 10-15 iterations are usually required.

To insure the elimination of round-off, the matrix solution routine is written in double precision on the IBM 360.

2.3 Generation of Tabular Data

A significant amount of data must be input in tabular form to the program. Provisions have been made in the program to simplify this input by using default data and fitting functions similar to the ones in FLARE.

A default set of burnups are given in the code and are the values 0, 4, 8, 12, ..., 48 GWD/MTU. A default set of axial leakage factors are also built into the code for a 12 foot PWR. These are presented in Figure 2.1. Corrections for cores other than 12 foot are made by the expression:

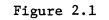
BIAS =
$$\left(\frac{H+0.5}{12.5}\right) \left(BIAS(12') - 1.0\right) + 1.0$$
 (2.15)

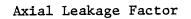
where

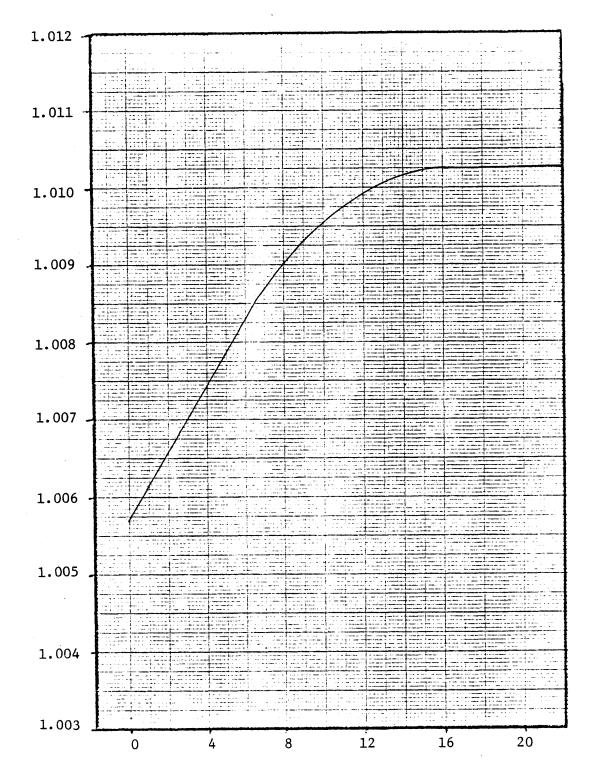
H is the core height in feet and 0.5 feet is the approximate reflector savings. (This is about 7.6 cm for each end of the core).

The k_{∞} may be entered in tabular form or by coefficients of a function. When they are input in tabular form, care must be taken to ensure that the different sets for different enrichments have the same burnup correspondence. The k_{∞} entered should correspond to a full power k_{∞} including equilibrium xenon, but should not include any soluble poison.

When FLARE type input is used to generate this data, the functional form may be used. The function is given by:

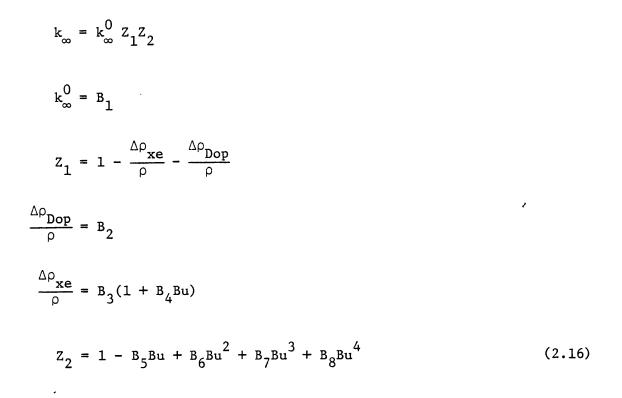






Region Burnup (GWD/MTM)

Axial Leakage Factor



where Bu is burnup in GWD/MTU and ρ is the reactivity for the different variables (xenon, doppler).

The burnable poison worth as a function of burnup is also entered. The nominal BP worth is the worth of one rod in one assembly. This can be tabulated or entered in a functional form. The function form is given by

$$BP(Bu) = B_1(1 + B_2Bu + B_3Bu^2 + B_4Bu^3 + B_5Bu^4)$$
(2.17)

where Bu is the burnup in GWD/MTU. It is usually a fairly good approximation to assume that the BP worth is relatively insensitive to the enrichment.

If a dataset for a fuel cost calculation is to be prepared, then isotopic data as a function of burnup and enrichment is required. The data needed are the uranium depletion, U-235 enrichment and fissile plutonium production. This information may be input or standard default data will be used. This default data is the same as that built into the MUDEL (1,5) code and consists of three empirical relations. For a given burnup (GWD/MTU) and enrichment (w/o) the following equations are used.

$$u/U_{o} = \exp[-0.00159 \text{ Bu}(1 - 4.1 \times 10^{-3} \text{ Bu})]$$
 (2.18)

$$U_{25} = \varepsilon \exp[-0.1162 \frac{Bu}{\varepsilon^{0.965}} (1 + 2.75 \times 10^{-5} Bu^2 \varepsilon)]$$
(2.19)

$$10^{3} \text{ Pu/U}_{o} = 4.795 \ \varepsilon^{0.4} \ \left[1 - \exp(-0.1425 \ \frac{B_{4}}{\varepsilon^{0.6}})\right]$$
 (2.20)

The accuracy of these expressions are discussed in Reference (1) and will not be reproduced here.

.

2.4 Generation of Shuffle

The program can on option automatically determine the shuffling scheme from cycle i-1 to cycle i. Two options are presently available and they deal only with the treatment of the center assembly. The shuffling is done very systematically. The feed fuel is assumed in the highest number regions. Then the regions from the previous cycle are inserted starting from the highest number region in the previous cycle. All the assemblies in each region are used except for the last region (excluding the center assembly) which uses only the number of assemblies needed. When there is a center assembly of a different region, two options are available. The first option uses an assembly from the previous cycle.

Explicit shuffling schemes must be input if the regions are not to be progressed in sequential order or if assemblies are to be reinserted after sitting out of the core for one or more cycles.

2.5 Generation of Edge Data

FLAC is different from all other programs in the need for edge data. That is the number of assembly edges adjacent to assemblies of each region and the periphery. Given a loading pattern, one can carefully count and generate this information. However, one of the attributes of FLAC is the ability to perform future cycle calculations without requiring loading patterns. Therefore, coding was generated to enable the automatic computation of this edge data for two different loading pattern schemes. The two schemes are commonly referred to as the ring pattern and the checkerboard pattern. The ring pattern assumes that the regions are located in symmetric

rings starting with the first region at the center. The checkerboard pattern assumes that the highest number regions form a peripheral ring with the interior regions arranged in a checkerboard distribution. The coding developed will not duplicate the edge data for a specific loading pattern, but it is general coding which will closely approximate the information for any shuffling procedure which is used.

Both schemes use the same procedure for the peripheral assemblies. The only information needed is the number of assemblies across the core and the number of assemblies on the periphery with edges on the exterior. The code assumes that the highest number region will be furthest out in the core. Therefore, it starts with those assemblies with two edges on the exterior, then goes to those assemblies with one edge on the exterior and then those assemblies on the periphery at a corner with no edges on the exterior. The number of edges on the exterior is simply:

$$n_{\alpha} = 4N_{row}$$
(2.21)

where N is the number of assembly rows. The number of peripheral assemblies with two edges on the exterior is given by:

$$n_{2\alpha} = n_{\alpha} - N_{p} \tag{2.22}$$

where N_p is the number of assemblies on the periphery. The number of assemblies at an inside corner of the periphery with no edges on the exterior is given by:

$$n_{0\alpha} = n_{\alpha} - N_{p} - 4 \tag{2.23}$$

and of course the number of assemblies in the peripheral region with one edge on the exterior is given by:

$$n_{1\alpha} = n_{\alpha} - n_{2\alpha} \tag{2.24}$$

The program begins by segregating the peripheral assemblies with two edges on the exterior. They are assumed to have no common edges. The program then segregates out the peripheral assemblies with one edge on the exterior. Edges are assumed to exist between the assemblies with one and two edges on the exterior. However, a maximum is set of one edge for those assemblies with one edge on the exterior and two edges for those assemblies with two edges on the exterior. The assemblies on the periphery with one edge on the exterior should by definition only have one edge left to face the interior. Therefore, any excess edges are assumed to be between those assemblies themselves.

The program next fills in the inside corner assemblies. These assemblies have two edges adjacent to peripheral assemblies and two edges on the interior. Therefore, starting with any excess edges left from the assemblies with two edges on the exterior the correspondence is assigned to have two edges of each interior corner assembly adjacent to some peripheral assembly. This completes the core periphery.

The checkerboard pattern is generated assuming a uniform distribution of the interior regions. If there is a region with a center assembly, it is arbitrarily assumed to adjoin region 3 assemblies. The peripheral assemblies are assumed to adjoin the interior regions in direct proportion to the number of assemblies in each region. Similarly, the interior

regions are assumed to be adjacent to the other interior regions in direct proportion to the number of assemblies in each region. Any remaining edges are assumed to be adjacent to assemblies of its same region. The logic for the checkerboard is ideal and therefore will be an approximation to any actual loading pattern.

The ring pattern on the interior is a little more complex. One must first define the rings. A standard definition is used which gives the number of assemblies within a ring and all smaller rings. The rings are numbered sequentially from the center with each additional row of assemblies being a new ring. Also, different formulas were developed for an even number of assemblies versus an odd number of assemblies in a core.

For an odd number of assemblies, the number of assemblies with a ring is given by:

$$N_{I} = 2(I-1)(I+2) + 1 \qquad I = 1, 2, 3, \dots$$
(2.25)

This was found to give a very accurate estimate of the number of assemblies in each ring.

For an even number of assemblies, no direct formula was found, so an area approach was used:

$$N_T = \pi I^2$$
 $I = 1, 2, 3, ...$ (2.26)

which was then rounded off to the nearest eight assemblies (or four for the inner-most ring).

In determining the edge correspondence between rings, many variables have to be considered. It was assumed that assemblies in ring I would first fill any vacant edge positions in ring I-1. But only one edge per assembly was allowed to be adjacent to ring I-1. Then, each ring must have eight additional edges on the outside of the ring since it is larger. Any edges left over are assumed to be within assemblies in the same ring. In meshing with the periphery, it is treated as any other ring interface.

2.6 Interpolation Scheme

Many of the values used by the program are input in tabular form and require interpolation or extrapolation. Whenever possible, the interpolation scheme used is the Lagrange 3 point interpolation. The interpolation of $f(x_i)$ given a fixed value of x is given by:

$$f(x) = \alpha_1 f(x_i) + \alpha_2 f(x_{i+1}) + \alpha_3 f(x_{i+2})$$
(2.27)

where

$$\alpha_{1} = \frac{(x_{i+1} - x)(x_{i+2} - x)}{(x_{i+1} - x_{i})(x_{i+2} - x_{i})}$$

$$\alpha_{2} = \frac{(x_{i} - x)(x_{i+2} - x)}{(x_{i} - x_{i+1})(x_{i+2} - x_{i+1})}$$
and
$$\alpha_{3} = \frac{(x_{i} - x)(x_{i+1} - x)}{(x_{i} - x_{i+2})(x_{i+1} - x_{i+2})}$$

The values x_i , x_{i+1} and x_{i+2} are chosen so that the value x is encompassed.

For extrapolation, a simple linear model is used to assure that no change in slope occurs in the function being extrapolated. The linear extrapolation is given by:

$$f(x) = \alpha_1 f(x_i) + \alpha_2 f(x_{i+1})$$
(2.28)

where

$$\alpha_{1} = \frac{(x_{i+1} - x)}{(x_{i+1} - x_{i})}$$

$$\alpha_2 = 1 - \alpha_1$$

The values x_i and x_{i+1} are the closest to the desired variable x. When the independent variable is burnup, it is assumed that the table is monotonically increasing and that there are at least three burnups tabulated. The fit will be made using one burnup below the desired value and two points above it, if possible.

When the independent variable is enrichment, a more complicated scheme is used. The interpolation is made only over those of the same "fuel type" as the desired enrichment. Thus it is possible to separate different fuel types such as plutonium of different recycle modes, thorium, stainless clad, larger hydrogen to uranium ratio, etc. The program makes no implicit assumption on the number of enrichments available for interpolation or any sequential order. The program will try to determine three points in the table - the closest value greater than the desired value, the closest value less than the desired value and the next closest value to the desired value on either side. This will ensure the best three points to use in the 3 point Lagrange interpolation. Note that for extrapolation or when only two enrichments of the same type are given, only two points are found and a linear extrapolation is used. When only one enrichment is in the table, a constant is generated independent of enrichment.

To insure proper interpolation, care must be taken in generating the table. Enrichments of the same type should not be within 0.10 w/o of each other. For close enrichments, secondary variables may distort the true relationship with enrichment.

When the interpolation is to be made over both enrichment and burnup, a two-step procedure is used. The three enrichments and three burnups which are to be used in the interpolation are found as described above. Then for each reference enrichment, an interpolation is made to yield the dependent variable at the specified burnup for that reference enrichment. Then the second step consists of interpolating the values at the desired burnup and reference enrichments to give the value at the desired burnup and enrichment.

2.7 <u>Burnup Calculation</u>

To determine cycle lifetime a burnup calculation must be performed. Three different options are available - straight depletion with no control, straight depletion with uniform control and a Haling calculation.

The three different options all require the beginning of cycle region burnups. These are determined by input or by the shuffle from the previous cycles. The end of cycle burnups from the previous cycles are modified,

if necessary, to reflect partial discharge as discussed in Section 2.8 and used as the beginning of cycle burnups for the current cycle.

The straight depletion options start with the beginning of cycle burnups, calculate relative power sharings and then incrementing the region burnups by the burnup step size times the relative region power sharings. The nominal step size in the program is set at 4 GWD/MTU with a shorter time step taken last to give the exact desired cycle lifetime. A large time step is used since the variation of power sharings with burnup in a cycle is usually small on a region-wise basis. Therefore, the error is small.

For the no control depletion, the interpolated k_{∞} are used directly without assuming any form of control. With uniform control, a search is performed each depletion step to determine the uniform change in k_{∞} to be applied to each region to yield the desired k_{eff} of 1.0 within \pm .00001. The uniform control option is used when the boron worth is input.

With a Haling (6) calculation option, an iteration is performed to give the end of cycle power distribution which is held constant for the entire cycle depletion. Thus an initial power distribution is assumed; the EOC region-wise burnups are then determined using these power sharings; and then the EOC power distribution is calculated. This new power distribution is then used to recalculate the region-wise EOC burnups. This procedure is used iteratively until the maximum region-wise power sharing difference between iterations is less than 0.001.

Thus, the Haling option assumes that the reactor will be controlled so that a constant power shape will be maintained during the core life. Since end of life implies that there is no excess reactivity; there is no control present. Therefore, the Haling calculation assumes that the power

distribution during the cycle is identical to the end of cycle power distribution which has no control.

2.8 Discharge Burnups

The number of assemblies discharged in any cycle is equal to the total number of assemblies in the region minus the total number of assemblies used from that region in all future cycles. It is assumed that the burnup of all assemblies used in future cycles are identical and that the discharge burnup of each assembly discharged is identical. But an option exists to allow for different burnups between discharged and kept assemblies. It is assumed that the distribution of burnups in a region is uniform over a specified range. Also, it is assumed that the highest average burnup assemblies will be discharged and the lowest burnup assemblies will be discharged and the lowest burnups within a range of $Bu_{avg} \pm \Delta Bu$, then the burnups are given by:

$$Bu^{\text{Discharged}} = Bu_{\text{avg}} + \Delta Bu \left(1 - \frac{\# \text{Discharged}}{\# \text{Region}}\right)$$
(2.29)

$$Bu^{Kept} = Bu - \Delta Bu \left(\frac{\# \text{ Discharged}}{\# \text{ Region}}\right)$$
(2.30)

where the numbers are the number of assemblies discharged or in the region. A typical value of ΔBu is 2 GWD/MTU.

2.9 Burnable Poison Treatment

The burnable poisons are treated on a region-wise basis. For each cycle the total number of BPs are given and the fraction in each region.

This allows for a search on the number of BP rods in the whole core if desired. The depletion of the BP rods is handled on a functional basis of the region average burnup. Thus the BP rods are not actually depleted. The worth in one assembly of a single BP rod is tabulated as a function of region burnup. The code assumes a linear relationship between worth and the number of BP rods. This is a good assumption for a small number of BP rods/assembly. Therefore, to determine the change in k_{eff} for a region due to burnable poisons, the following relationship is used:

$$k_{eff}(w BP) = k_{eff}(wo BP) - \frac{(No. BPs)(Fraction in Region)}{No. Assemblies in Region} \rho_{BP}(Bu)$$
(2.31)

where $\rho_{\rm BP}({\rm Bu})$ is the worth of one BP rod at a burnup Bu. Note that the BP worth is determined at the region average burnup, so there is an implicit function that fresh BPs are only placed in fresh fuel.

2.10 Cycle Search Procedure

It is possible to search for any one of a number of variables each cycle. The allowed search variables are:

cycle burnup number of burnable poison rods EOC k_{eff} enrichment of any feed region

These searches are all performed using a Newton-Raphson procedure. The new value of the variable y for iteration i+1 is given by:

$$y^{i+1} = y^{i} - slope (k_{eff}^{i} - k_{eff}^{want})$$

$$(2.32)$$

$$y^{i} = y^{i-1}$$

slope =
$$\frac{y^2 - y^2}{k_{eff}^{i} - k_{eff}^{i-1}}$$

Convergence is assumed when $|y^{i+1} - y^i| < \varepsilon$ where ε is .001 for cycle burnups, 1.0 for number of BP rods, and .01 for enrichment.

These convergence criteria were set to yield acceptable results without unnecessary accuracy. They also allow for significant changes in the k_{eff} to insure that division by the difference in k_{eff} does not produce overflow problems in the computer. Note that no iteration is needed for the EOC k_{eff} , this is just a straight forward calculation. To eliminate possibly large variations in feed enrichment, allowable minimum and maximum values of enrichment are input. When the enrichment hits the edge of this band, it is set to the value and the k_{eff} calculated.

2.11 Convergence Criteria

The program uses many different iterative scheme to calculate the desired result. In some cases there can be a significant nesting of iterative searches. Up to five iterations nestled together is possible. These different iterative schemes are located in three different subroutines and have built in convergence criteria. The various convergence criteria are listed in Table 2.1 along with the subroutine where it may be found.

TABLE 2.1

CONVERGENCE CRITERIA

Routine	Parameters	Max. Iterations	Epsilon
EQCALC	EOC burnups	50	0.01
CYCALC) EQCALC)	Cycle burnup	50	0.001
CYCALC) EQCALC	Number of BP rods	50	1.0
CYCALC) EQCALC	Feed enrichment	50	0.001
CYCALC	Boron defect	-	0.0001
CYCALC	EOC Haling burnups	50	0.001
SOLVE	{Eigenvector {Eigenvalue	50	0.001 0.00001

.

CHAPTER 3

VERIFICATION AND QUALIFICATION

The checkout of a computer program consists of two separate and distinct parts - verification and qualification. The verification of a program consists of demonstrating that the program performs the calculations as specified. The qualification of a program consists of determining the applicability of the program for its use to calculate various parameters using a prescribed method. Thus verification consists of demonstrating that the program does what it is supposed to do and qualification consists of showing that what it does gives good results using certain methods.

Sensitivity studies were also performed to determine how accurate the input needs to be to give acceptable results. These studies demonstrate the adequacy of one value for the albedo and the absorption probability (w) and the adequacy of the automatic generation of the edge data.

3.1 Verification

There are a number of areas which can be separately verified. The two basic areas are the actual solution of the matrix equation and the interpolation of data. These two aspects form the basis of the program. All the other aspects can be verified by successful execution of the program. For example, the various search procedures must converge to a correct solution or be wrong.

3.1.1 Matrix Verification

A 3 x 3 matrix was chosen for verification since it can readily be solved by hand. For a 3 x 3 matrix given by

$$A = \begin{pmatrix} \beta_{1} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \beta_{2} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \beta_{3} \end{pmatrix}$$
(3.1)

The eigenvalues are found by solving the equation det $(A - \lambda I) = 0$. Writing out the determinant gives:

$$\lambda^{3} - (\beta_{1} + \beta_{2} + \beta_{3})\lambda^{2} + (\beta_{1}\beta_{2} + \beta_{1}\beta_{3} + \beta_{2}\beta_{3} - \gamma_{13}\gamma_{31} - \gamma_{23}\gamma_{32} - \gamma_{12}\gamma_{21})\lambda + (\beta_{1}\gamma_{23}\gamma_{32} + \beta_{2}\gamma_{13}\gamma_{31} + \beta_{3}\gamma_{12}\gamma_{21} - \beta_{1}\beta_{2}\beta_{3} - \gamma_{12}\gamma_{23}\gamma_{31} - \gamma_{13}\gamma_{21}\gamma_{32}) = 0$$
(3.2)

The largest eigenvalue is found by solving the above third order polynomial. This can be done easily by an iterative search. Given the eigenvalue, the eigenvector is then calculated by:

$$S_{1} = 1.0$$

$$S_{2} = \frac{\lambda - \beta_{1}}{\gamma_{12}} - \left(\frac{\gamma_{13}}{\gamma_{12}}\right) S_{3}$$

$$S_{3} = \left[\frac{\lambda - \beta_{1}}{\gamma_{12}} - \frac{\gamma_{21}}{\lambda - \beta_{2}}\right] / \left[\frac{\gamma_{23}}{\lambda - \beta_{2}} + \frac{\gamma_{13}}{\gamma_{12}}\right]$$
(3.3)

A sample case was set up to test the matrix solution procedure. The problem was a three region problem as specified in Table 3-1. The values of α and w used are:

w = 0.10 and $\alpha = 0.30$

The corresponding matrix for this problem as defined in Equations 2.9, 2.10 and 2.12 is:

60234	.34855	.053005
.389791	.66066	.050650
.066852	.056948	.92880

The cubic polynomial giving the eigenvalues of this matrix using Equation 3.2 is:

$$\lambda^{3} - 2.191797\lambda^{2} + 1.428722\lambda - 0.2416968 = 0$$
 (3.4)

which gives as a solution $\lambda = 1.0527$ within the accuracy of a hand calculator. Applying Equation 3.3 gives the corresponding eigenvector:

$$\underline{S} = \begin{pmatrix} 1.0 \\ 1.131 \\ 1.0589 \end{pmatrix}$$

MATRIX SOLUTION TEST PROBLEM

.

•

.

		Fractio	on of Edges	Adjacent t	o Region
Region	k_eff_	1	2	<u>3</u>	Exterior
1	1.0039	0.0	0.8680	0.1320	0.0
2	1.1011	0.8850	0.0	0.1150	0.0
3	1.2380	0.1350	0.1150	0.4620	0.2880

The answer calculated by the program was

$$\lambda * = 1.0527 \text{ and } \underline{S}^* = \begin{pmatrix} .940 \\ 1.06 \\ .999 \end{pmatrix}$$

where $\lambda * = \lambda$ and <u>S</u>* = .940 <u>S</u>.

Note that since S*, S are eigenvectors they can be normalized in any fashion. Hence the fact that S* = 0.94S implies equality within a multiple and hence implies that S and S* are identical eigenvectors. Therefore, the program is correctly computing the matrix and its largest eigenvalue and its corresponding eigenvector.

3.1.2 Interpolation Verification

The interpolation scheme was also verified. From the data in Table 3.2, the k_∞ for 3.20 w/o at a burnup of 22.0 GWD/MTU was calculated. The program interpolated value was 1.1011.

As described in Section 2.6, the interpolation is done at the specified burnup for each enrichment and then the value for the specified enrichment is calculated. From equation (2.27) we get the relationship:

$$k_{\infty}(22) = -.083333k_{\infty}(8) + 1.020833k_{\infty}(20) + .0625k_{\infty}(36)$$
(3.5)

so

 $k_{\infty}^{2.9 \text{ w/o}}(22) = 1.07506$ $k_{\infty}^{3.1 \text{ w/o}}(22) = 1.09243$

DATA FOR \mathbf{k}_∞ INTERPOLATION

Burnup (GWD/MTU)	k _∞ (2.90 w/o)	k _∞ (3.10 w/o)
1	1.31136	1.32605
8	1.22296	1.23898
20	1.09447	1.11178
36	.95529	.97183

.

. .

From Equation (2.28) we also have that

$$f^{3\cdot 2}(22) = -0.5 f^{2\cdot 9}(22) + 1.5 f^{3\cdot 1}(22)$$

$$f^{3\cdot 2}(22) = 1.1011$$
(3.6)

Thus the hand calculation verifies the computer calculation for the interpolation and extrapolation.

3.2 Qualification

The verification demonstrates that the FLAC program is functioning as formulated. It remains to be demonstrated that the program is qualified for use in fuel management studies. In order to evaluate the effectiveness of a computer program, a large number of varied cases should be run. The work done by Rieck (10) in his doctoral thesis is well suited to provide benchmark cases for the FLAC code qualification. A large number of variations in the number of feed assemblies for the Zion reactor was investigated. Zion is a 4 loop PWR reactor by Westinghouse. Its core consists of 193 assemblies distributed in 15 rows. The information needed to generate the assembly ${\bf k}_{\infty}$ data consistently with Rieck's data is available in his thesis. Based on current knowledge, I believe that the data presented by Rieck is not in perfect agreement with current calculations for the Zion core. However, the use of the same ${\bf k}_{\infty}$ data assures that the comparison is meaningful. A comparison using the same basic input as was used with more precise methods demonstrates how closely FLAC reproduces the more precise calculations. As with any computer code, the accuracy of the input will determine the accuracy of the output to a large degree. What the qualification demonstrates is the ability of FLAC to obtain essentially the same results as a more precise tool using the same basic input.

Two depletion cases were run for eight cycles - one feeding 64 assemblies at 3.20 w/o and the other feeding 48 assemblies at 3.76 w/o. Then various perturbation calculations were performed for cycle 9 with variations in the number of feed assemblies from 48 to 84 and significant variations in enrichment. There were 19 change cases starting from the 64 feed assembly case and 6 change cases starting from the 48 feed assembly case.

The model employed in FLAC used the k_{∞} as generated from the SIMULATE fit as used by Rieck. The automated generation of edge data with a checkerboard interior and the automated shuffling option were used. The number of feed assemblies and the feed enrichment were input and the program was allowed to calculate the cycle lifetime. A boron search was performed and a discharge burnup variation parameter of 2.0 GWD/MTM was used. The desired end of life k_{eff} was 1.0001 which corresponds to approximately 10 ppm of residual boron which would translate into about 0.01 GWD/MTU. The results of the comparison are presented in Tables 3.3 through 3.6.

As can be seen from comparing the results, the FLAC calculations agree very well with the more detailed calculations performed by Rieck. The results for all 39 cycles yield an average difference in the cycle burnup of only 79 MWD/MTU (less than 1% error) and a standard deviation of 104 MWD/MTU. This is excellent accuracy for a survey tool since the cycle average burnup is seldom known by even the best methods to within \pm 100 MWD/MTM. This is due to the fact that there does exist manufacturing, design, enrichment and cross section uncertainties all of which create uncertainty in the final design calculations.

- 41

	3-D Calcu	ulations	FL	FLAC		Difference	
<u>Cycle</u>	Cycle <u>Burnup</u> (GWD/MTU)	Discharge Burnup (GWD/MTU)	Cycle <u>Burnup</u> (GWD/MTU)	Discharge Burnup (GWD/MTU)	Cycle <u>Burnup</u> (GWD/MTU)	Discharge Burnup (GWD/MTU)	
1	15.600	16.943	*	17.407	-	.207	
2	9.652	25.115	9.715	25.800	.063	.685	
3	9.894	32.076	10.095	31.672	.201	404	
4	10.284	30.306	10.164	30.331	120	.025	
5	10.038	30.401	10.050	30.502	.012	.101	
6	10.084	30.419	10.104	30.471	.020	.052	
7	10.081	30.399	10.093	30.456	.012	.057	
8	10.081	30.400	10.090	. 30.408	.009	.008	
			Avera	age .	.028	.091	

ZION 64 FEED ASSEMBLY CYCLE STUDY

Standard Deviation .088 .280

*Because of burnable poisons, the search was on the calculated k eff at end of life.

TABLE	3.4

	3-D Calcu	ulations	FLA	AC	Differ	
	Cycle	Discharge	Cycle	Discharge	Cycle	Discharge
Cycle	Burnup	Burnup	Burnup	Burnup	Burnup (GWD/MTU)	Burnup (GWD/MTU)
	(GWD/MTU)	(GWD/MTU)	(GWD/MTU)	(GWD/MTU)	(GWD/MIU)	(GWD/MIO)
1	15.600	17.174	*	17.899	_	.725
2	8.228	23.917	8.436	25.394	.208	1.477
2	8.887	31.559	9.419	32.224	.532	.665
3	0.00/	31.339	9.419	J2•227	.552	
4	9.868	~ 37.029	9.780	38.239	088	1.210
					0.71	1.61
5	10.151	38.433	10.422	38.897	.271	.464
6	9.659	39.191	9.731	39.710	.072	.519
0	.055	37.171	<i></i>	0,0,0,0	•••	
7	9.769	39.488	9.892	40.092	.123	.604
			0.040	20.000	100	506
8	9.811 .	39.370	9.943	39.892	.132	.526
*						
	Ave	rage .			.179	.774
	Sta	ndard Deviat:	ion		.178	.344

ZION 48 FEED ASSEMBLY CYCLE STUDY

*Because of burnable poisons, the search was on the calculated ${\rm k}_{\rm eff}$ at end of life

Case	Number	Enrichment (w/o U-235)	3-D <u>Burnup</u> (GWD/MTU)	FLAC <u>Burnup</u> (GWD/MTU)	Difference (GWD/MTU)
1	56	4.42	10.953	11.049	.096
2	60	3.86	10.709	10.785	.076
3	64	3.70	11.057	11.138	.081
4	68	3.40	11.007	11.063	.056
	72	3.30	11.279	11.321	.042
6	76	3.06	11.104	11.146	.042
7	80	2.94	11.174	11.211	.037
8	60	4.34	11.606	11.666	.060
. 9	64	3.96	11.618	11.639	.021
10.	68	3.78	11.874	11.873	001
11	64	4.20	12.097	12.123	.026
12	68	3.88	12.088	12.086	002
13	76	3.42	12.103	12.106	.003
14	68	4.34	13.100	13.115	.015
15	72	4.00	13.019	12.955	064
16	80	3.60	13.118	13.087	031
17	72	4.50	14.127	14.225	.098
18	76	4.20	14.131	14.096	035
19	84	3.76	14.150	14.076	074

ZION VARIATION FROM 64 FEED ASSEMBLIES

Average

+25.6

Standard Deviation

48.6

.

Case	Number Fed	Enrichment (w/o U-235)	3-D Burnup (GWD/MTU)	FLAC Burnup (GWD/MTU)	Difference (GWD/MTU)
1	56	4.06	11.780	11.918	.138
2	60	3.74	11.861	12.007	.146
3	68	3.30	12.023	12.230	.207
4	64	4.38	13.823	13.950	.127
5	68	4.06	13.850	13.914	.064
6	76	3.52	13.822	13.826	.004

1

ZION VARIATION FROM 48 FEED ASSEMBLIES

Average	114.3

Standard Deviation

64.6

These cases covered a very broad range of conditions going from 48 feed assemblies to 84 feed assemblies in a variety of loading patterns. These cases also covered successive cycles and demonstrated that the deviations do not become larger cummulatively. Instead they demonstrate convergence as the equilibrium cycling scheme is approached. Yet the edge data was generated in FLAC automatically for each case, thus demonstrating the adequacy of this procedure for predicting average cycle burnups.

A comparison of the region discharge burnups in Tables 3.3 and 3.4 also shows good agreement. As expected, the average discharge burnup variation in the three region core is approximately three times the cycle burnup variation and the average discharge burnup variation in the four region core is approximately four times the cycle burnup variation. The largest discrepancies appear to be associated with regions 1, 2 and 3 which were all in cycle 1 with burnable poisons. Very little work was done with the burnable poison treatment and it is expected that future work could refine this calculation. Also, there are large discrepancies in the four region case for cycles 2 through 4 where partial regions were discharged. One primary reason for these differences is the method of selecting the assemblies to be discharged. FLAC seeks to optimize fuel utilization and hence discharges the highest burnup fraction of the region. In the comparison study the selection seems to be fairly random with in some cases the lowest burnup assembly in a region being discharged.

The actual power or burnup sharings between regions in a cycle is subject to more variation since it is highly dependent on the loading pattern. Table 3.7 shows the comparison between the actual burnup sharings from the explicit multi-dimensional calculation and the FLAC calculation for an equilibrium cycle in Zion with both 48 and 64 feed assemblies. As can be

BURNUP SHARING COMPARISON

64 Feed Assemblies

Region	3-D Burnup <u>Sharing</u> (GWD/MTU)	FLAC Burnup <u>Sharing</u> (GWD/MTU)	Number of Assemblies
1	8.2	8.2	1
2	8.3	8.7	64
3	10.8	10.4	64
4	10.5	10.7	64

48 Feed Assemblies

Region	3-D Burnup <u>Sharing</u> (GWD/MTU)	FLAC Burnup <u>Sharing</u> (GWD/MTU)	Number of Assemblies
1	7.5	6.8	1
2	7.9	8.6	48
3	10.1	8.9	48
4	12.3	11.8	48
5	9.9	10.0	48

.

seen the results are generally within \pm 10% and for the 64 feed assembly, the results are excellent.

Based on these comparisons, FLAC appears to generate acceptable results for long range fuel management studies. The cycle length will be estimated within \pm 500 MWD/MTU with a high degree of certainty assuming the input is generated properly.

3.3 Sensitivity Analysis

FLAC is an approximate method. It is not a detailed and sophisticated procedure to yield very accurate results. Therefore, it does not require extremely accurate input data. Some of the input data does vary somewhat with particular core configurations (i.e., W and ALPHA, the absorption probability and the albedo), yet only average values are used. Other parameters will vary depending on the actual loading pattern (number of edges), yet only a general type of loading pattern is assumed. This section will discuss the various analyses which were done to quantify the sensitivity to these variables.

3.3.1 Sensitivity to W and ALPHA

A 193 assembly core, 3 region case was calculated for different combinations of values of W, the probability that a neutron born in one node will be absorbed in an adjacent node, and of ALPHA, the albedo. The range of values was W from 0.100 to 0.140 and ALPHA from 0.25 to 0.50. The values that have been found to give the best results for many different cases are W equal to 0.115 and ALPHA equal to 0.30. The sensitivity analysis demonstrates that variation in the parameters by \pm 10% yields results with acceptable accuracy (\pm .005 $\Delta\rho$ and \pm 5% in regionwise power). These results

are graphically displayed in Figures 3.1 through 3.8. These figures indicate that the inner regions are relatively insensitive to the variations. Thus the major variations are the peripheral region power sharing and the k_{eff} .

3.3.2 Sensitivity to Edge Data

Cycle length and power sharings are affected by the loading pattern choice, yet when running FLAC one does not want to have to know the precise loading pattern in each cycle. One would like the results to be relatively insensitive to the loading pattern, while being capable of determining the relative merits of different types of loading pattern schemes. And it appears that FLAC can do this.

First, the results are relatively independent of the variations in a loading pattern within a particular scheme. All of the verification runs used the automatic checkerboard loading edge data routine built into the code. Thus the verification included random deviations from the actual loading pattern assuming the general constraint of a peripheral feed with a checkerboard interior.

A separate study was performed with edge data corresponding to various loading patterns. The data corresponds to the actual loading pattern in a three region core using a checkerboard interior and automatically generated for a checkerboard pattern, a ring pattern and checkerboard patterns with burnt fuel in the periphery. The results are summarized in Table 3.8. As can be seen from this table, there are only minor differences between the actual checkerboard pattern and the code generated one, yet significant differences can be noticed between the other loading patterns. For example, the cases with the burnt assemblies

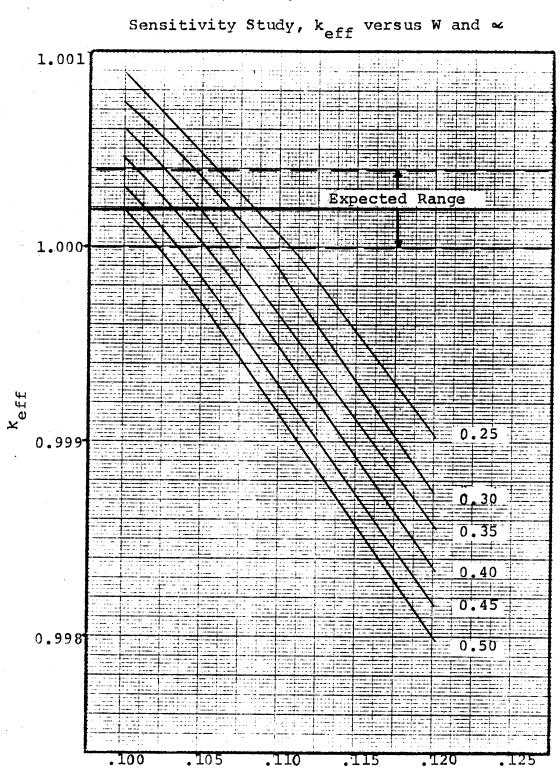
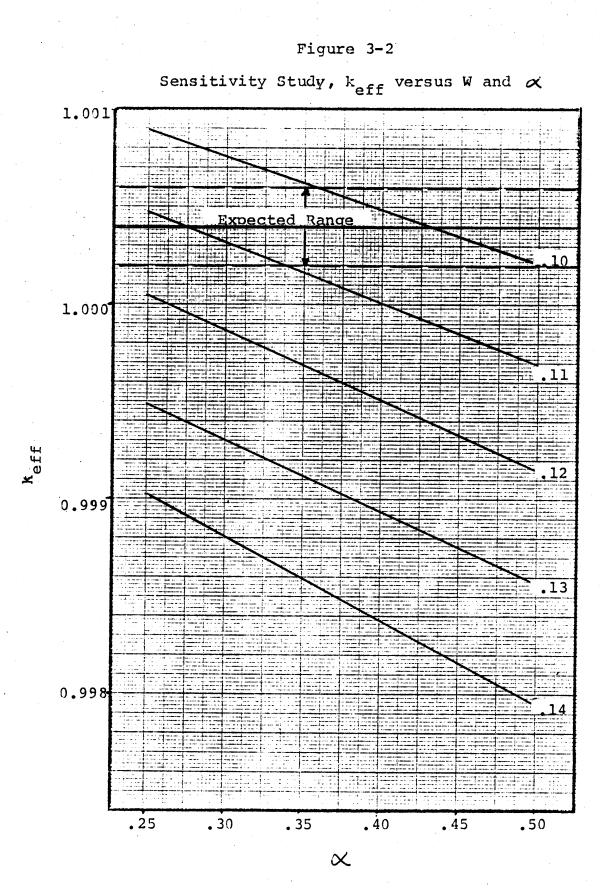


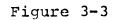
Figure 3-1

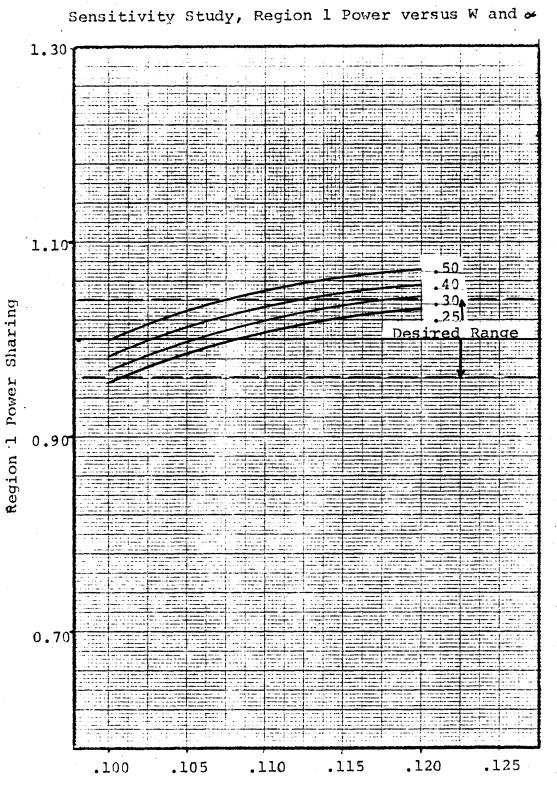
50

W



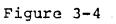


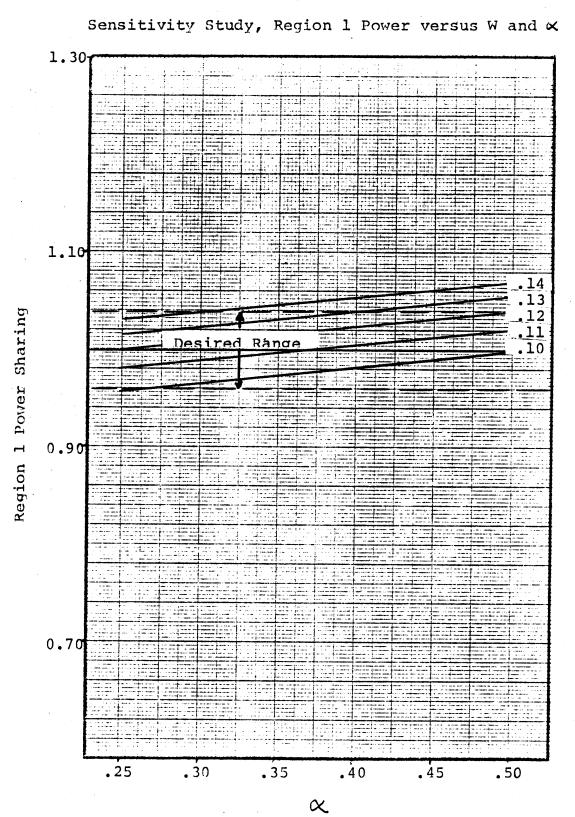




52·

W





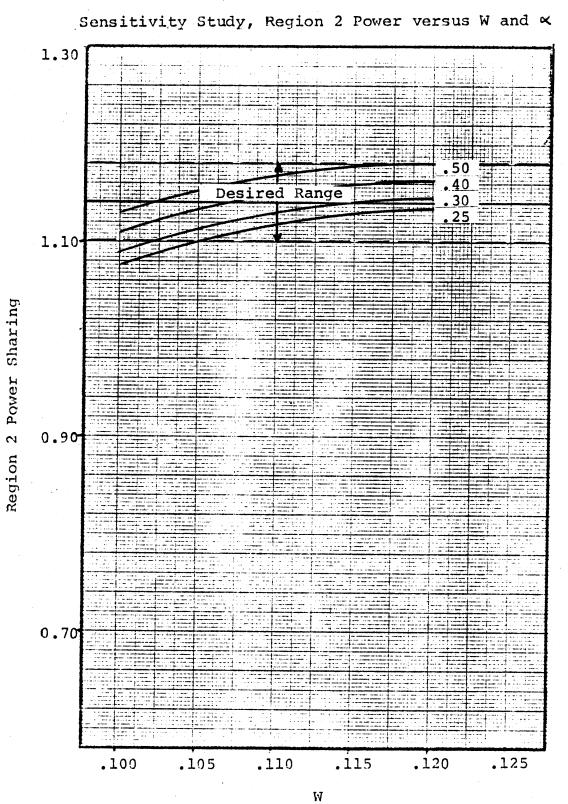


Figure 3-5

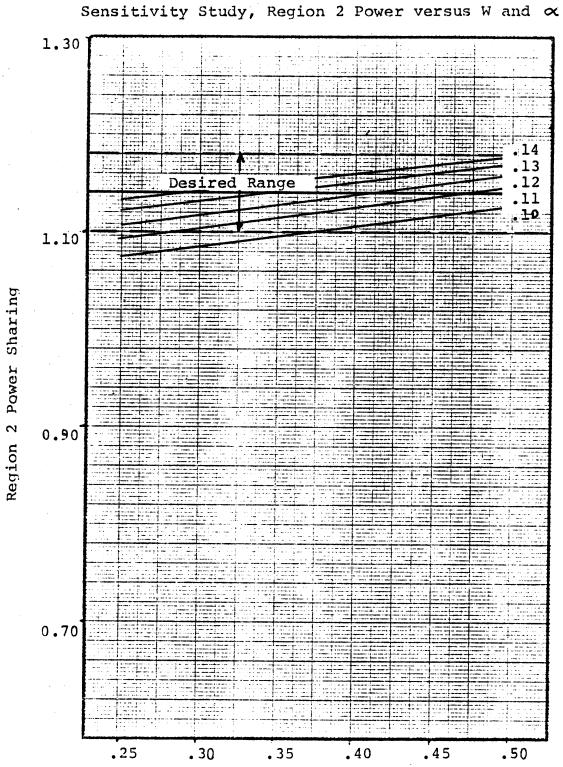
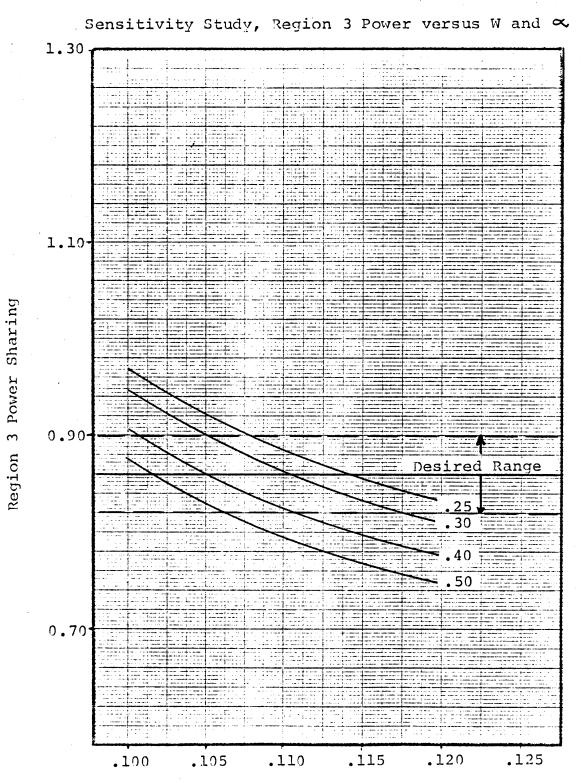
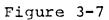


Figure 3-6

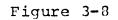
 \propto

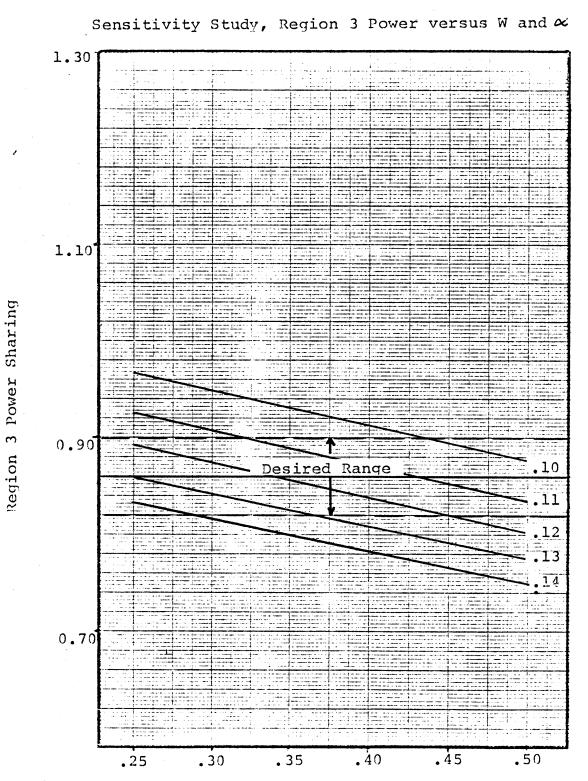




56

N





 \propto

LOADING PATTERN SENSITIVITIES

	Actual Checkerboard	Automatic Checkerboard	Automatic Ring	Checkerboard 12 Burnt on Periphery	Checkerboard 24 Burnt on Periphery
Cycle Length (GWD/MTU)	11.01	10.98	10.40	11.09	11.62
Power Sharings					
Reg 3 BOL	.933	.943	1.122	1.154	1.284
EOL	.900	.905	.953	1.063	1.176
Reg 2 BOL	1.110	1.106	1.262	1.058	1.047
EOL	1.108	1.106	1.169	1.086	1.091
Reg 3 BOL	.961	.954	.626	.792	.674
EOL	.994	.991	.883	.852	.736

Based on: 157 Assembly Core 52 Feed Assemblies at 3.20 w/o U-235 Equilibrium Cycle

.

on the periphery show increased cycle lifetime due to reduced neutron leakage. However, these cases also indicate higher powers in the feed region. The ring model agrees with the general view that the checkerboard patterns are superior to ring patterns for larger cores since rings have higher peaking and less cycle lifetime.

A second study was done using the Zion model as discussed in Section 3.2. Equilibrium cycle calculations were done to determine the feed enrichment required for a fixed 11.0 GWD/MTM cycle length for various numbers of feed assemblies from 48 to 84. These cases were run with a checkerboard interior and a ring interior. These results are summarized in Table 3.9 and the required enrichment given in Figure 3.9. These results also demonstrate the superiority of the checkerboard pattern. It also indicates the variation in power sharings as the number of feed assemblies is changed. This type of data is presently impossible to obtain without explicit multi-dimensional calculations.

Therefore, FLAC presents a good compromise on the need for accurate knowledge of a loading pattern. It can differentiate between different basic types but does not need a detailed description of the loading pattern.

3.3.3 Sensitivity to Boron Search

An option exists in FLAC to calculate a uniform poison worth each step and use this to modify the region k_{∞} 's. This is comparable to a search to criticality using the soluble boron concentration as the search variable. Two calculations, one with and one without the search, were performed to determine the effect of the search. The results are listed in Table 3.10. These show that the effect is fairly small with the largest effect at

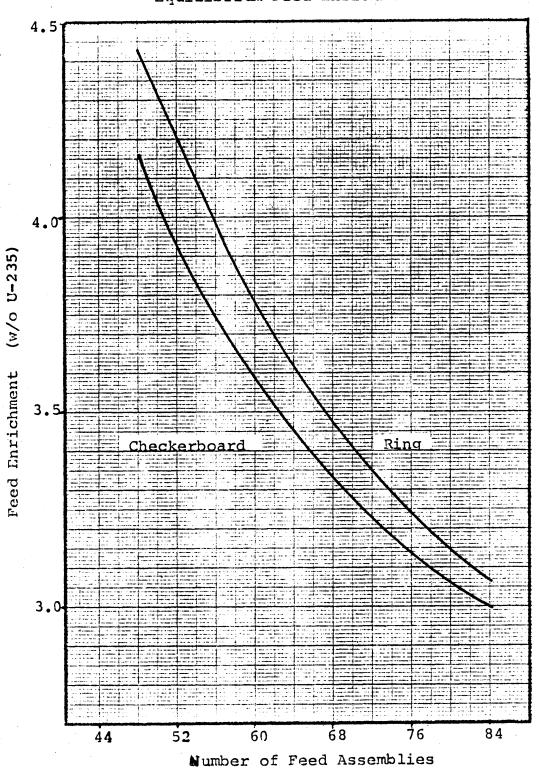
EQUILIBRIUM FEED ENRICHMENT

	Checkerboard	Pattern	Ring Pat	tern
Number of	Feed	Peak	Feed	Peak
Feed Assemblies	Enrichment	Power	Enrichment	Power
	(w/o U-235)		(w/o U-235)	
48	4.133	1.207	4.424	1.651
52	3.943	1.165	4.216	1.477
56	3.822	1.308	3.987	1.481
60	3.615	1.216	3.780	1.407
64	3.443	1.161	3.619	1.391
68	3.339	1.148	3.503	1.402
72	3.243	1.147	3.367	1.343
76	3.156	1.141	3.246	1.311
80	2.070	1.123	3.145	1.289
84	2.987	1.097	3.052	1.284

`**.**

.

Based on: 193 Assembly Core 11.0 GWD/MTM Cycle Length Equilibrium Cycle



Equilibrium Feed Enrichment

Figure 3-9

SENSITIVITY TO BORON SEARCH

Parameter	Boron Search	No. Search
EOL k eff	0.9992	1.0002
Power Sharings		
Region 3 BOL	.908	.882
EOL	.886	.887
Region 2 BOL	1.121	1.128
EOL	1.114	1.114
Region 1 BOL	.975	.993
EOL	1.002	1.001

Based on: 157 assembly core 52 Feed Assemblies at 3.2 w/o U-235 Cycle length of 11.00 GWD/MTU Equilibrium Cycle

•

.

beginning of life where the poison concentration is the highest. However, at end of life the effect is very small. The effect of the boron search is within the uncertainty of the calculation and its use is optional. However, to assure consistant comparison of different cases; the same option should be used in each case.

CHAPTER 4

CONCLUSIONS AND RECOMMENDATIONS

4.1 Recommended Method

FLAC is an approximate tool giving results only within a specified range. Care must be taken to avoid losing sight of this basic characteristic. When preparing the input to FLAC, therefore, one should not get buried in details which in the end will have no significant affect on the results.

The interpolation scheme in FLAC gives very good results over fairly wide ranges of enrichment. Therefore, I would recommend enrichments spaced at intervals of about 0.5 w/o U-235. In fact, data corresponding to two enrichments very close in enrichment can cause errors since there are usually secondary parameters which vary from region to region in addition to the enrichment. And for very small changes in enrichment, these secondary changes could be quite significant. As far as burnup variations, steps of 4.0 GWD/MTM are generally sufficient and even larger steps would probably be adequate.

The actual k_{∞} data can be taken from LEOPARD (2) calculations which properly homogenize the whole assembly. The k_{∞} should correspond to full power fuel temperatures and moderator densities, contain equilibrium xenon, samarium and the other fission products and not contain any soluble boron.

The isotopic information can be generated using the built in fit unless significant variation is seen or unless a different fuel type (i.e. thorium) is used.

The use of the automated shuffle option and the automated edge data generation option are highly recommended. There will probably be more cases

to explicitly input the shuffle than the edge data. For example, when assemblies are reinserted into the core after sitting out one or more cycles or when different shuffling schemes are to be tried which do not have the feed region exclusively in the periphery. However, since most loading patterns are variations of a checkerboard loading pattern, this option is highly recommended.

The use of the boron search option feature is optional since its effect is small. However, not using this option does slightly simplify the the input.

Since the results are relatively insensitive to the values of W and ALPHA, it is recommended that the default values be used for standard PWR cores. For BWR cores or non-standard cores, a study should be performed to determine the optimum values of these parameters.

For the search procedure, it is recommended that the burnup search be used most frequently. The enrichment search should be used mainly in equilibrium cycle calculations where large fluctuations in enrichment will not occur.

The one big assumption in generating the FLAC equations from the FLARE equations is the homogeneity of the region. It assumed that the k_{∞} is identical for each assembly in the region. This assumes that the burnup and enrichment are the same for each assembly within a region. This is usually not possible, but regions as used by the code should be assemblies having the same enrichment and approximately the same burnup at beginning and end of life. Thus split enrichment feeds should be divided up as well as regions placed both on the periphery and in the interior.

The use of the above methods should produce good results with a minimum of effort. Their use is therefore recommended.

4.2 Method Accuracy

Based on the verification, qualification and sensitivity analysis performed, the recommended method will produce good results. In long range fuel management there are many variables which can alter the results of a study. Many of these variables are external and can not be controlled. Therefore, it is superfulous to calculate any of the parameters to a high degree of accuracy.

The FLAC program using the recommended procedure can predict the cycle length to within \pm 500 MWD/MTM to a high degree of confidence. Likewise, it can predict the region power sharings to within \pm 10%. A higher degree of accuracy is possible when parameter variations are made for comparisons of their relative effects. This degree of accuracy is sufficient for almost all long range fuel management studies.

4.3 Conclusions

A new long range fuel management program, FLAC, has been developed and its use demonstrated. The program has been written, verified and qualified for fuel management studies. The program can differentiate between the various types of loading patterns on a generic basis and thus is a significant improvement over the weighted k procedures which have been used in the past. However, variations within loading pattern types have been shown not to be significant, hence explicit loading patterns are not required which simplifies the fuel management study considerably compared to multidimensional depletion calculations. Therefore, FLAC satisfies the original goal of combining the best attributes of the k weighting procedures and the multi-dimensional calculations for an optimum long-range fuel management tool.

4.4 Future Work

The development of this method allows for an extensive amount of future work possibilities. The program can be used for an endless number of studies and significant improvements can be made to the program to make it more usable. Some of the basic schemes for adding to the program are fuel cycle cost calculations, burnable poison search procedures and generation of data to verify the loading pattern-cycle length compatability with design considerations.

Presently the program prepares a data set for use in fuel cycle cost calculations. This could be expanded to actually include a fuel cycle cost calculation like MITCOST built into the program. This would greatly simplify the comparison of various proposed fuel management schemes since it would give a direct comparison of fuel cycle costs. This addition could be made relatively easily since the data set is presently prepared. It would require the input of the fuel cost parameters and the coupling to the fuel cost program.

The feasibility of adding options concerning the burnable poisons is intriguing. Since burnable poisons are handled on a region-wise basis it is possible to determine the affect of the region-wise distribution of the burnable poisons in the core. Therefore, it is conceivable that a search procedure could be built into the program to search for the burnable poison distribution which produces the optimum power distribution. Of course, this raises difficulties as to what time in the cycle one wishes to optimize the power distribution and also how to effectively converge on a desired shape. A Haling method may produce good results for the power distribution optimization; that is to compute iteratively a BOC power distribution with

BPs that have the same power distribution as the EOC with depleted BPs.

Another search feature with burnable poisons is the possibility of searching for the correct number to give an acceptable moderator temperature coefficient at the beginning of the cycle. This requires the computation of the BOC critical boron concentration and using some procedure to generate the appropriate moderator temperature coefficient. This scheme is straightforward and should cause no convergence problems. However, the optimum distribution of these burnable poisons may have to be addressed. Thus, these burnable poison search features may have to all be added at once.

The design considerations offer a third different area for program expansion. This area is multifold, since there are many varied design considerations to evaluate. They all require some sort of fits or functions to extrapolate the program data into the controlling variables. These fits may be used to give various coefficients or control rod worths or correction terms to the region power sharings may be possible to generate peak rod powers and burnups. Some of the variables that could possibly be checked are:

moderator temperature coefficients soluble boron coefficients power coefficients and defects total control rod worth maximum $F_{\Delta H}$ for a rod maximum assembly burnup maximum pellet burnup delayed neutron fraction and all other various quantities.

It is conceivable that the majority of the design constraints can be roughly estimated for many future cycles. This would allow one to see if the fuel management of one cycle has any long range problems.

The program in its present form can be used for a number of studies. One possibility is the usefulness for BWR fuel management. The program was designed to be able to calculate BWR or PWR cores. However, all the qualification effort was for PWRs. Therefore, methods for using FLAC for BWR fuel management have yet to be formulated and qualified.

The structure of the program allows for feasibility studies to be performed in areas where only advanced and expensive tools could previously be used. For example, cycling studies using many burnable poisons such as 18 month cycles or the in-out-in fuel management strategy. FLAC has the capability of investigating the effects on power distribution and cycle lifetime capability due to the number and placement of burnable poisons. Also, the effects of different burnable poison compositions (i.e. different w/o of B_4C in AlO₃) could be relatively easily studied. Here one has the trade off of rods depleting quickly or slowly and its affect on residual worth and cycle power sharings.

The program can also be used to determine the equilibrium affect of a particular fuel management scheme very easily. The sophisticated methods require numerous cycles in succession to reach a pseudo equilibrium cycle. FLAC has the equilibrium cycle calculation as an option. Thus one can easily determine the equilibrium effect of different types of loading patterns or of splitting the feed region up into various batches of different enrichments. The possibilities are endless, but most could be handled easily by FLAC.

CHAPTER 5

REFERENCES

- Antonopoulos, P. T. and Turnage, J. C., "Management Analysis of Nuclear Allocation for the Generation of Electricity", Nuclear Technology, Vol. 34, p. 347 (August, 1977)
- Barry, R. F., "The Revised LEOPARD-4 Code Spectrum Dependent Non-Spatial Depletion Program", WCAP-2579, Westinghouse Electric Corporation (March, 1965)
- 3. Cadwell, et.al., "PDO-7 Reference Manual", WAPD-TM-678, Westinghouse Electric Corporation (January, 1967)
- 4. Delp, D. L. et.al., "FLARE A Three-Dimensional Boiling Water Reactor Simulator", GEAP-4598, General Electric Company (1964)
- 5. Forbes, I. A., et.al., "A Rapid Empirical Method for Multi-cycle Fuel Cost Analysis", Trans. Am. Nucl. Soc., Vol. 21, p. 257 (1975)
- 6. Haling, R. K., "Operating Strategy for Maintaining an Optimum Power Distribution Throughout Life", TID-7672, USAEC, (1964)
- Lang, F. D. and Isakson, R. A., "Computer Application of Cyclic Reactivity Carryover Effects in Fuel Management", Trans. Am. Nucl. Soc., Vol. 13, p. 39 (1970)
- 8. Naft, B. N., "The Effect of Regionwise Power Sharing on PWR Incore Fuel Management", Trans. Am. Nucl. Soc., Vol. 21, p. 655 (1972)
- 9. Pilat, E. E., personal communication.
- Rieck, T. A., "The Effect of Refueling Decisions and Engineering Constraints on the Fuel Management for a Pressurized Water Reactor", Ph.D. Thesis, MIT Department of Nuclear Engineering, February, 1974
- VerPlanck, D. M. and Ferguson, D. R., "SIMULATE, Reactor Simulator Code, User's Manual", Yankee Atomic Electric Company, (1972); revision (September, 1975)
- 12. Vogt, D. K. and Carlson, R. W., "A Model for Rapid Evaluation of Fuel Management Options", Trans. Am. Nucl. Soc., Vol. 22, p. 319 (1975)

APPENDIX A

FLAC USER'S MANUAL

FLAC was written to be an easy to use program that could be readily adopted by a utility engineer, vendor engineer or graduate student to his own specific needs. However, to allow flexibility in its use, many options had to be provided. This tends to make the input look quite expansive; where it is very trivial for most cases. Many cases are as small as five cards, while it is possible to generate some relatively large input decks.

Similarly, the output was kept to a minimum to keep the engineer from becoming lost in a maze of numbers. The relavent data is presented in a neat, well labeled format with only digits with some significance editted. It seems rather unnecessary to give the k_{eff} to the nearest tenth of a pcm when it is probably only accurate to the nearest 200 pcm. However, to allow meaningful perturbation calculations, the k_{∞} is given . to the nearest 10 pcm.

A.1 Input Preparation

A.1.1 Input Description

The input to FLAC is relatively straight forward and simple. All the input is in fixed format, so particular attention must be paid to positioning of the data on the card. Cases are easily stacked and much of the information can propagate from case to case.

The input can be divided into two distinct parts - the general case data and the cycle by cycle data. The input parameters are all listed in Table A.1 with their corresponding descriptions. This section will

TABLE A-1

INPUT PARAMETERS

COLS. PARAM. TYPE SUBPT UNITS DESCRIPTION

i.

***	TITLE CARD	(20A4)	
1-80	TITLE A	(20)	FULL CARD OF TITLE DATA FOR CASE
* * * *	CASE DATA C	ARD A (2413)	
1- 3	NASS I		NUMBER OF ASSEMBLIES IN WHOLE CORE
4- 0	NADIAM 1		NUMBER OF ASSEMBLIES ACROSS CORE
7- 9	NBU I		NUMBER OF BURNUPS IN INTERPOLATION TABLE. IF NOT ENTERED, USE DEFAULT OR PREVIOUS CASE BURNUPS.
			IF =-1, THEN TABLES INPUT AS COEPFICIENTS OF DEFINED FUNCTIONS
			AS IN SIMULATE.
10-12	NRICH I		NUMBER OF K INF. SETS TO BE READ IN. IF =0, USE DEFAULT OR PREVIOUS CASE DATA, IF >0, REPLACE PREVIOUS
			SET WITH NEW SET. IF <0, ADD THIS MANY NEW SETS TO PREVIOUS SETS.
13-15	150 1		IF =1, READ IN ISOTOPIC DATA ALONG WITH K INF. DATA. IF =-1, generate isotopic data prom
			MUDDLE FIT.

16-18 IFCOST	I	 	IF =1, PREPARE DATA SET FOR A FUEL CYCLE COST CALCULATION.
19-21 INFIT	L	 	IF =1, READ IN FIT DATA.
22-24 IHAL	I	 . .	IF =1, A HALING CALCULATION WILL BE DONE FOR EACH CYCLE.
25-27 DEBUG	L	 ·	IF =1, DEBUG PRINT OPTION
28-30 NW	Ţ		NUMBER OF DIFFERENT VALUES OF W TO USE. IF NOT ENTERED, SET TO 1.
31-33 NA	I	 	NUMBER OF DIFFERENT VALUES OF ALPHA TO USE. IF NOT ENTERED, SET TO 1.

**** CASE	DATA CA	RD B (12F6.	.2)
1-6 ₩	ĸ	 	PROBABILITÝ THAT A NEUTRON BORN IN AN ASSEMBLY WILL BE ABSORBED IN AN ADJACENT ASSEMBLY. DEFAULT = 0.115.
7-12 ALPH	A R		ALBEDO ON EXTERIOR EDGE OF CORE. DEFAULT = 0.30.
13-18 ELIM	k	(1) W/O	MINIAUM FEED ENRICHMENT IN SEARCH
19-24 ELIM	R	(2) W/O	MAXIMUM FEED ENRICHMENT IN SEARCH IF NOT INPUT, SET TO 4.5
25-30 BUDI	F F	GWD/MT	A BURNUP DIFFERENCE FOR SPLIT REGION BURNUP OF SPLIT REGION IS GIVEN BY BU = AVG.BU - BUDIF*(1-N KEPT)/N)
31-36 CYBI	AS R		K EFF BIAS FACTOR - DIVIDES INTO K
37-42 HEIG	HT K	PLET	CORE HEIGHT FOR GENERATION OF AXIAL BIAS FACTOR AS A FCN OF BURNUP.
43-48 ULCA	D R	MTM	TOTAL CORE AVERAGE LOADING.
49-54 POWE	RR	MW T	TOTAL CORE POWER.
55-60 DELW	k		INCREMENTAL CHANGE IN W TO BE USED TO GENERATE NW-1 VALUES OF W AFTER THE BASE INPUT VALUE.
61-66 DELA	R		INCREMENTAL CHANGE IN ALPHA TO BE USED TO GENERATE NA-1 VALUES OF ALPHA AFTER THE BASE INPUT VALUE.

**** BURNUP DATA CARDS (12P6.2) (ONLY IF NBU>0)

BURNUP R (I) GWD/MTM BURNUP ARRAY IN ORDER OF INCREASING BURNUP. 12 ITEMS PER CARD, UP TO 20 ENTRIES READ IN. ONLY READ IN IF NBU>0, THEN NBU ENTRIES READ

- **** AXIAL BIAS CARDS (6E12.5) (ONLY IF NBU>O, HEIGHT=O) BIAS R (I) -- AXIAL BIAS FACTOR AS A FUNCTION BU, ENTERED ONLY IF NBU>O AND HEIGHT=O. NBU ENTRIES CORRESPONDING TO BURNUP(I), 6 ENTRIES PER CARD.
- **** BP WORTH CARDS (6E12.5) (ONLY IF NBU>0) BP R (I) PCM WORTH OF ONE BP ROD/ ASSEMBLY AS A FUNCTION OF BURNUP(I). ENTERED ONLY IF NBU>0. NBU ENTRIES, 6 PER CARD.

* * * *	ВЪ	COEFFICI	ENT CARD	(6 E 1	2.5)	(ONI	Y IF	ияд<0)		
	в	R	(I)		COEFFICIE						
					OF ONE BI	P ROD/A	SEMB	LY. P	UNCT	ION	IS
					B (1) * (1. (B (4) *B(**2	+ . 1
					WHERE BU				•	/MTU	•

- **** ENRICHMENT CARDS (1286.2) (ONLY IF NRICH NE 0)
 - ENRICH K (I) W/U ENRICHMENT CORRESPONDING TO EACH DATA SET. W/O U-235 OR W/O PU. NRICH VALUES ENTERED, 12 PER CARD.
- **** FUEL TYPE CARDS (1216) (ONLY IF NRICH NE 0) ITYPE I' (1) -- FUEL TYPE CORRESPONDING TO EACH DATA SET. NRICH VALUES ENTERED 12 PER CARD.

**** POWER/FISSION RATIO CARDS (12F6.0) (ONLY IF NRICH NE 0) POWFIS K (I) MEV/N POWER TO FISSION SOURCE RATIO FOR EACH FUEL TYPE. (KAPPA/NU) NRICH VALUES, 12 PER CARD.

**** K INF CARDS (6E12.5) (ONLY IF NRICH NE 0, NBU GE 0)

KINF	R	(I,J)	 K INF COR	RBS	PONDIN	G TO	BURNUI	P (I)	
			FOR ENRICI	HME	NT J.	NBU	VALUES	FOR	
			EACH SET,	6	VALUES	PER	CARD,	NRICH	
			SETS.						

**** KINF COEFFICIENT CARDS (6E12.5) (ONLY IF NRICH NE 0, NBU<0)

В	8 (I) ·	COEFFICIENTS OF FUNCTION FOR K INF.
		AS A FUNCTION OF BURNUP. ONE SET
		ENTERED FOR EACH ENRICHMENT.
		THE FUNCTION IS OF THE FORM:
		K INP = B(1) * Z 1 * Z 2
		Z1=1.0-B(2) - B(3)*(1.0+B(4)*BU)
		$Z_2=1.0 - B(5) * BU + B(6) * BU * 2 +$
		B (7) *BU **3 + B (8) *BU **4
		WHERE BU IS BURNUP IN GWD/MTU.
		NOTE THAT B(1) TO B(8) ARE TO BE
		ENTERED, SO 2 CARDS/SET ARE NEEDED.

**** ISOTOPIC DATA CARDS (6E12.5) (ONLY IF ISO=1, NRICH NE 0)

ISOTOP R (1, J, 1) KG/MTM URANIUM CONTENT AS A FUNCTION OF BURNUP (1). NBU ENTRIES, 6 ENTRIES PER CARD FOR ENRICH (J).

ISOTOP R (I,J,2) KG/MTM U-235 CONTENT AS A FUNCTION OF BURNUP(I). NBU ENTRIES, 6 ENTRIES PER CARD FOR ENRICH(J).

ISOTOP R (I,J,3)KG/MTM TOTAL PU CONTENT AS A FUNCTION OF BURNUP(I). NBU ENTRIES, 6 ENTRIES PER CARD FOR ENRICH(J).

ISOTOP R (I,J,4) -- FISSILE/(TOTAL PU) RATIO AS A FUNCTION OF BURNUP(I) FOR ENRICH(J).

NGTE *** THE ORDER OF CARDS ARE ALL BURNUPS IN SEQUENCE ON CARD(S) TO FORM A SET. THEN EACH SET FOR DIFFERENT ISOTOPIC INFORMATION (4 SETS) FOR ENRICH(J) TO FORM A LARGER SET WHICH IS THEN REPEATED FOR EACH ENRICHMENT. IF NOT ENTERED, DEFAULT DATA WILL BE USED FROM FIT. DATA IS ONLY USED FOR COST DATA SET PREPARATION.

* * * *	COEFFIC	IENT	FIT	DATA SET	(5F6.2/ (6E12.5)/) (ONLY IF INFIT =1)
1- 6	XEWRTI	R		PCM	TOTAL HFP XENON WORTH AT BOL
7-12	DPWRTH	R		PC M	TOTAL DOPPLER DEFECT HFP TO HZP AT BOL.
13-18	BWRTH	ĸ		PCM/PPM	BURON WORTH AT BOL.
19-24	TEMP	R	(1)	DEG P	HZP TEMPERATURE.
25-30	TEMP	R	(2)	DEG F	AVERAGE CORE HFP TEMPERATURE.
	COFIT	R	(1)		MODERATOR TEMPERATURE FIT PARAMETERS UP TO 10 ALLOWED, 6 PER CARD.

**** CYCLE DATA **** ALL OF THE REMAINING DATA IS INPUT FOR EACH CYCLE, THEN FOR THE NEXT CYCLE, ETC. THE CYCLES SHOULD BE IN INCREASING ORDER. ONE MUST START WITH CYCLE 1, BUT BOC EURNUPS MAY BE ENTERED. NO CYCLE MAY BE ELIMINATED FROM THE SEQUENCE, EXCEPT AT THE END. THUS ANY NUMBER OF CYCLES UP TO 20 MAY BE FUN AND THEY MUST BE CONSECUTIVE. CYCLE 21 IS AN EQUILIBRIUM CYCLE AND IS SEPARATE FROM THE REST. THE CYCLE DATA MUST BE FOLLOWED BY A BLANK CARD AFTER THE LAST CYCLE INPUT.

****	CYCLE OPTION	CARD	(8I3,6F6.0)
1- 3	15 I		CYCLE NUMBER 1-20 OR 21 FOR EQUIL. CYCLE.
4- o	ISURCH I	(15)	SEARCH PARAMETER FOR CYCLE IS. =1 SEARCH ON CYCLE BURNUP =2 SEARCH ON NUMBER OF BPS =3 DETERMINE EOL K EFF. =10+J SEARCH ON ENRICHMENT FOR FEED J IN CYCLE IS
7- 9	15007 1	(15)	SHUFFLE GENERATION OPTION =0 INPUT SHUFFLE FOR THIS CYCLE =-1 USE SHUFFLE FROM LAST CASE FOR THIS CYCLE. >0 GENERATE SHUFFLE DATA AUTOMATICALLY =1 ASSUME PERITHERAL FEED AND CENTER ASSEMBLY FROM PREVIOUS CYCLE. =2 ASSUME PERIPHERAL FEED AND CENTER ASSEMBLY FROM CYCLE 1.

10-12	NREG	ï	(IS)		NUMBER OF REGIONS IN CYCLE (<16)
13-15	NFEED	I	(IS)		NUMBER OF FEED REGIONS IN CYCLE. NAY BE 0,1,2, OR 3.
16-18	IBOC	I		· · · · · · · · · · · · · · · · · · ·	 =1 IF BOC REGION BURNUPS ARE TO BE ENTERED FOR THIS CYCLE. =0 IF BOC REGION BURNUPS ARE TO BE CALCULATED USING SHUFFLE DATA AND EOC REGION BURNUPS FROM PREVIOUS CYCLES. =-1 IF PREVIOUS CASE BOC REGION BURNUPS ARE TO BE USED THIS CYCLE.
19-21	1 Fedge	1	(15)		=0 INPUT THE EDGE DATA. =-1 USE THE EDGE DATA FROM THE PREVIOUS CASE FOR THIS CYCLE. >0 GENERATE EDGE DATA AUTOMATICALLY =1 ASSUME A RING GEOMETRY IN THE CORE FOR THE EDGE DATA. =2 ASSUME A CHECKERBOARD GEOMETRY FOR THE INTERIOR OF THE CORE.
22-24	DUMMY	I.			
25-30	CYLU	R	(IS)	GWD/M TM	DESIRED (OR GUESSED) CYCLE BURNUP.
31-36	Bb	R	(1S)		NUMBER OF BP RODS (OR GUESS) IN CORE.
37-42	BOLK	Ŕ	(15)		DESIRED END OF LIPE K EFF FOR CYCLE. CAN BE VARIED TO CHANGE FOR EOL BORON CONCENTRATION, COASTDOWN OR EARLY SHUTDOWN.

43-48 CAPPAC	K	(15)	PERCENT	CAPACITY FACTOR FOR CYCLE, EXCLUDING REFUELING SHUTDOWN.
49-54 CITIM	Ħ	(15)	DAYS	CYCLE LENGTH, EXCLUDING SHUTDOWN. ONLY 2 OF 3 (CYBU, CAPFAC, CYTIM) NEED BE ENTERED. IF 3 ENTERED, CYTIM IS RECALCULATED.
55-60 SDLEN	Ŕ	(IS)		REFUELING SHUTDOWN LENGTH. IF NOT ENTERED. SET TO 42.

* * * *	FEED DA	Ϋ́Λ	INPUT F REGIONS	POR 1 5 FOI 1ENT	A3,I3,2F6.0) EACH REGION IN CYCLE 1 OR JUST THE FEED R OTHER CYCLES. SPART WITH THE LOWEST REGION AND PUT IN ORDER OF INCREASING
1- 3	NAFEED	1	(IS,I)		NUMBER OF FLED I (OR REGION I IN
	NAC1	1	(I)		CYCLE 1) ASSEMBLIES.
4- 6	FTYPE	1	(IS,I)		FUEL TYPE FOR FEED I (OR REGION I
	C 1TYPE	I	(1)		FUEL TYPE FOR FEED I (OR REGION I IN CYCLE 1) CORRESPONDING TO INPUT
					TABLE OF K INF. INTERPOLATION WILL
•					ONLY BE DONE WITH IDENTICAL FUEL TYPES.
7- 9	FID	·A	(1, 21)		ID FOR FEED REGION 2. (IE. 5A,6PU).
	C11D	A	(1)		ID FOR FEED REGION 2. (IE. 5A,6PU). THIS IS USED FOR LABELING OF OUTPUT.
10-12	DUMMY	1			
13-18	FRICH	K	(IS,I)	W/0	ENRICHMENT OF FEED I IN CYCLE IS OR
			(I)		
19-24	FLOAD	R	(IS,I)	M TM	LOADING PER ASSEMBLY POÀ FEED I IN
	C 1LOAD				CYCLE IS OR REGION I IN CYCLE 1. IF NOT ENTERED, SET TO ULOAD/NASS.

* * * *	AEGION	SIZE	CARD	(1216)	(ONLY	IF	ISHUP(IS)=0)
---------	--------	------	------	--------	-------	----	--------------

NAREG I (I,IS) -- NUMBER OF ASSEMBLIES IN EACH REGION STARTING WITH THE HIGHEST BURNUP (INNERMOST) REGION. NREG(IS) ENTRIES, 12 PER CARD.

**** SHUFFLE CARD (1216) (ONLY IF ISHUF(IS)=0)

ISHUF I (I,IS) -- SHUFPLE INICIES SPECIPYING PREVIOUS LOCATION OF EACH REGION(I) IN CYCLE IS. VALUE OF FORM 100*K+L WHERE K IS THE PREVIOUS CYCLE NUMBER AND L IS THE REGION NUMBER IN THE CYCLE. FOR A FEED REGION, K=O AND L IS THE FEED NUMBER (1,2, OR 3) NREG(IS) ENTRIES, 12 PER CARD.

* * * *	EDGE CAR	DS (1116)		(ONLY IF IFEDGE(IS)=0)
1- 6	JR	1		REGION NUMBER CORRESPONDING TO DATA ON THE WHOLE CARD.
7-12	NEDGE	I (1,K)	حت هشه	SPECIFICATIONS OF THE NUMBER OF EDGES
13-18	AEDGE	I (2,5)		ADJOINING REGION JR IN CYCLE IS.
19-24	NEDGE	I (1, K+1)		NEDCE(1,.) IS THE REGION NUMBER
25-30	NEDGE	I = (2, K+1)		ADJOINING REGION JR AND NEDGE (2)
	ETC.			IS THE NUMBER OF EDGES BETWEEN THEM.
				FOR EDGES ON THE EXTERIOR, SET
				NEDGE(1,.)=0. NOTE THAT K IS JUST
				A COUNTER. ALSO ONLY NON-ZERO
				NUMBER OF EDGES MUST BE ENTERED AND
				ONLY ONE SET FOR EACH REGION PAIR.

NOTE: LAST CARD WITH EDGE DATA SHOULD BE FOLLOWED BY A BLANK CARD.

5 PAIRS MAY BE ENTERED ON EACH CARD.

**** BP FRACTION CARD (12F6.2) (ONLY IF BP(IS) NE 0)

BPFRAC R (I,IS) -- FRACTION OF BP RODS IN EACH REGION I IN CYCLE IS. NREG(IS) ENTRIES, 12 ENTRIES PER CARD.

**** BOC BURNUP CARD (12F6.2) (ONLY IF IBOC=1) BOCBU R (I,IS)GWD/NTM BOC REGION BURNUPS FOR CYCLE IS. NREG (IS) ENTRIES, 12 ENTRIES / CARD.

**** LND OF CYCLE (IS) DATA - REPEAT SET FOR NEXT CYCLE. **** AFTER LAST CYCLE DATA SET, PUT A BLANK CARD. **** CASES MAY BE STACKED DIRECTLY.

elaborate on those items in Table A.1 which need further clarification or which are very important.

In the table, the variable name as used in the program is given along with the columns where it should be input. The type of the variable is also given (I for integer, R for real and A for alphanumeric). Note that no decimal points may be used with integer data, but that they should be used with real data. The subscripts, if any are given and the units appropriate to the code are also specified along with a description of the input quantity.

Sample cases are given in Section A.1.2 for an example. The cases are a base case with sequential cycles followed by two equilibrium cycle cases. These cases demonstrate many of the input options such as shuffle data, edge data and BOC region burnup input. The cases are marked up to identify the various parameters.

Every case begins with a title card. This card is used to title the output and as such appears at the top of each page.

The title card is followed by case data cards A and B. Card A contains integer data while card B contains real data. Only the first two parameters on cards A and B are required. The parameters NBU, NRICH, ISO and INFIT specify if specific sets of input are also to be entered. NBU and NRICH must be entered for the first case in a set of cases since the k_{∞} table must be input in some manner. The debug print option should be used only when necessary to determine the cause of an undetermined problem. This option prints a significant amount of intermediate data out chiefly through the subroutine BUGOUT which is a generalized routine for printing arrays.

The values of W and α on card B are the two most important quantities to be entered. W is the W_i parameter in the basic FLARE equation and α is

the albedo. W by definition is the probability that a neutron born in an assembly will be absorbed in a single adjacent assembly. W usually has a value in the range of 0.10 to 0.14 and α usually has a value in the range of 0.25 to 0.40. See Section 3.3.1 for a discussion on the sensitivity of these parameters.

If a sensitivity study on W and α is desired, the variables NW, NA, DELW and DELA should be entered on case data cards A and B. This will produce a NWxNA matrix of cases using different combinations of W and α . The first case uses the input values of W and α and succeeding cases change the relative values by DELW and DELA. Thus one can easily perform a sensitivity study on the combinations of W and α .

Also, on case data card B are the minimum and maximum enrichments allowed in an enrichment search. From experience, limits on the feed enrichment during a search are necessary to preclude wild fluctuations which could possibly occur in the feed enrichment. The variable BUDIF allows one to have a uniform burnup distribution over a specified burnup range to enable split regions to discharge a higher burnup than the region average. This option is discussed fully in Section 2.8. The cycle bias factor is an adjustment term which is applied to each cycle. Each k_{∞} is divided by this cycle bias factor. The core height is used to calculate the axial leakage effect using a default curve for 12 foot cores. The loading and power are used to convert from burnup units to time units and do not have to be entered unless appropriate cycle times are desired.

The burnup array is entered only on option. All tabular data uses this array as a base and therefore, all the data must be consistent with it. The default set is the data 0, 4, 8, ..., 48 GWD/MTU.

The BIAS input is generally used only to account for the effects of axial leakage effects as a function of burnup. However, it can be used as a general bias factor depending on the need. The region k_{∞} are divided by this bias factor.

The burnable poison worth can be entered in a functional form or in a tabular form. In either case the data is for the worth of one BP rod in a representative assembly as a function of burnup. This should include the worth of the poison, the displacement of moderator and the non-poison contribution such as clading or filler.

The input for the different fuel types is input next. Note that the data can add to existing data or create a new data set and also that the data can be input in tabular or functional form. The enrichment and fuel type are used to denote each different fuel data set. The enrichment can be used to denote the w/o of U-235, the w/o of plutonium or the w/o of U-233 or any other parameter (such as H/u ratio) which may want to be varied. It is best to maintain some deviation between enrichments when interpolation will be done. At least 0.1 w/o difference between enrichments of the same fuel type is recommended to insure that the secondary parameters do not enrichment. The fuel type allows one to differentiate between different fuel data sets where another parameter other than enrichment is important. This could correspond to different vendor's fuel, stainless versus zircaloy clad or different recycle batches of plutonium fuel. The power to fission ratio is important only for significantly different fuel types such as uranium and plutonium fuel. This corresponds to an average value of κ/ν , but since it is only a weighting function, any multiple will suffice. For

example, with all uranium fuel, an input of 1.00 for all enrichments is fine.

The k_{∞} data may be entered as coefficients in a polynomial function or in a tabular form. In any case the k_{∞} data should include xenon, samarium and full power doppler, but they should not contain any soluble boron.

The isotopic information can either be ignored, default generated or it can be input. The only use is to be interpolated at the discharge burnup for each batch for input to a fuel cost program. The data as input is interpolated, so the information should be in the desired end units.

The coefficient data is presently not used. The only input which has any effect is the boron worth entry which serves as a flag. When the boron worth is specified, each depletion step will include a search for a uniformly distributed poison to maintain core criticality.

Separate cycle data must be entered for each cycle for which a calculation is to be performed. Cycle 21 is an equilibrium cycle calculation and is independent of the other cycles. One can do just an equilibrium cycle, just a sequential series of cycles or both. The sequential series must always start with input for cycle 1 and include all cycles until the last one desired. The beginning of cycles region burnups may be entered for any cycle including the first. So cycle 1 may in fact be a later cycle with specified beginning of cycle region burnups. For cycle 1, no shuffle specification is needed, but the information for each region must be input. For succeeding cycles, only the information for the feed regions need to be entered. This information consists of the number of assemblies, an identifying ID, the fuel type and enrichment and the assembly average

loading if it is different from the reference core average.

The search parameter can vary between cycles. The search for the cycle burnup allows one to determine what burnup capability exists for a particular cycle. The search for the end of cycle k_{eff} is generally used for verification when the cycle burnup and feed enrichments are well known. The feed enrichment search allows the program to calculate the feed enrichment to produce the required cycle lifetime. This search procedure should be used with care since one is trying to influence the whole core with a change in only a fraction of the core. Large fluctuations in enrichment are possible.

The EOC k_{eff} which is desired is an input variable. This allows one to account for an early shutdown or a coastdown by inputting values respectively greater than 1.000 and less than 1.000.

The cycle burnup, cycle time and capacity factor are three variables; only two of which are independent. If any two are entered, the third is calculated. If all three are entered, the cycle time will be recalculated. Note that the cycle time and capacity factor do not include the refueling shutdown which is handled separately. The only variable which is actually used in the calculation is the cycle burnup. The other parameters are calculated only for editting and fuel cost input preparation.

The fuel shuffle can normally be calculated by the program. The program assumes that the assemblies are shuffled directly from the previous cycle. The feed regions are the highest number regions and the regions from the previous cycle are kept in the same order. The discharged assemblies are taken from region 1 first, then region 2, and so forth until the number of discharged assemblies in the previous cycle equals the number of feed assemblies. The automated shuffle will not work when

assemblies are being reinserted after at least one cycle in the spent fuel pit or when the lowest number region is not to be discharged. Also if the program is to calculate the edge data, it assumes that the regions are ordered in the manner such that the lowest number is at the center and the highest number is furthest out in the periphery. Therefore, if any assemblies other than feed assemblies are desired in the periphery, an explicit shuffle must be input.

When the shuffle is input both the number of assemblies in each region and where the region came from has to be entered. The description of the previous location is given by a number of the form 100 + K + J where K is the cycle number and J is the region number. For example, 903 implies that that region came from region 3 in cycle 9. The code will accept any cycle number less than the current cycle number and any number of assemblies may be transferred as long as it doesn't exceed the number available. For the equilibrium cycle, K must always be 21. The feed regions are also entered but in this case K is zero and J is the feed number (1, 2 or 3). If assemblies are to come from another unit, they should be treated as feed regions with the BOC burnup also input.

The edge data can also be generated automatically by the code. The edge data consists of the number of assembly sides of a region adjoining another specified region. The number of total edges is four times the number of assemblies in a region. Also, the number of edges between two regions is the same independent of the reference region. Some edges also are on the exterior of the core. The total number of these edges is four times the number of assembly rows across the core. The data is given by the reference region, then the adjoining regions and the number of common edges. The periphery is entered as region zero. A blank card is

entered to signify the end of the edge data information.

The beginning of cycle burnups may be entered on option. Any non-zero entries will over-ride any calculated burnups. Thus the burnups need only be entered for those regions whose burnups are to be specified. The other regions will use the calculated values. The beginning of cycle burnup are not used in an equilibrium cycle calculation, and hence their input will be ignored.

The cycle data is input on a cycle by cycle basis. A blank card is placed after the last cycle input to signify the end of the data for that case.

Cases may be stacked. The tabular data is saved from case to case and options exist to save some of the other data between cases. All other data must be reentered.

A.1.2 Sample Input

A small input deck has been generated to demonstrate the use of many of the various input options. The simple deck consists of five stacked cases and is listed on the following pages. These cases were also used to generate the sample output described in Section A.2.2.

The first case reads in the tabular data for three enrichments and four burnups. Generally, more burnup steps should be used, but for clarity only four were used in this case. The first case calculates the equilibrium cycle enrichment corresponding to a cycle length of 11.0 GWD/MTM and an end of cycle k_{eff} of 1.0010 which would correspond to residual soluble boron of 10 ppm. The edge data is input in this case.

	CASE - 3 LOOP, 3 REGION CHECKERBOARD CORE - ENRICH SEARCH
	3 -1 0 1
.115 .30	1005 .05
	20. 36.
1.0	1.0 1.0 1.0
900.	350. 25. 20.
2.5 2.9	3.1
• • • • •	
1.0 1.0	
	1.15416 1.02446 0.88846
	1.18096 1.05247 0.91329
1.28405	1.19698 1.06978 .92983
	-10.
21 11 1	4 1 0 0 0 11. 0.0 1.0010
52 U EQ	0 3.2
1	$\overline{3}$
2	3 180 4 28
	4 24
4	0 0 4 96
· · · · ·	U UU 4 90
FLAC TEST	CASE - 3 LOOP, 3 REGION CHECKERBOARD CORE - 3.10 %/0
157 15	
.115 .30	
	4 1 0 -1 0 11. 0. 1.001
52 0 EQ	
JZ U 1.2	
FLAC TEST	CASE - 3 LOOP, 3 REGION CHECKERBOARD - AUTO, 3.20 W/O
157 15	
.115 .30	
	4 1 0 2 0 11. 0. 1.001
52 U EQ	

FLAC TEST CASE - 3 LOOP, 3 REGION RING AUTO - 3.20 W/O 157 15 .115 .30 21 1 1 4 1 0 1 0 11. Û. 1.001 52 Û EQ Û 3.2 3 LOOP CORE - CYCLES 1-8 2.1/2.6/3.1 CY 1, 3.2 FEED - 3 REGION 157 15 0 0 0 0 0 0 0 .115 .30 2 0 14.5 900. 1.001 1 3 1 3 3 0 0 2.1 53 U 1 52 0 2 0 2.6 52 0 3.1 U J 0.0 0.80 0.20 2 0 11. 0 1.001 2 1 1 4 1 0 52 0 4 0 3.2 0 1.001 0 2 3 1 1 4 1 Û 11. 52 U 5 0 3.2 0 1.001 4 1 1 4 1 0 2 0 11. 52 0 6 0 3.2 5 0 2 0 1.001 1 4 1 1 0 11. 52 Û 7 0 3.2 0 1.001 6 1 1 4 1 0 2 Ù 11. 52 0 0 3.2 8 0 1.001 7 1 1 4 1 0 2 () 11. 52 0 9 0 3.2 8 1 1 4 1 0 2 0 11. 0 1.001 52 0 10 0 3.2

26

۰.

The second case is also an equilibrium cycle calculation, but the search is on cycle burnup given a 3.10 w/o feed. The edge data from the previous case is used in this case.

The third case is an equilibrium cycle searching on the cycle burnup, but with a feed of 3.20 w/o and the edge data generated automatically assuming a checkerboard interior.

The fourth case is the same as the third case except that a ring interior is assumed for the generation of the edge data.

The fifth case consists of cycles 1 through 8 starting with initial enrichments of 2.1, 2.6 and 3.1 w/o in cycle 1 and feeding 52 assemblies at 3.2 w/o each cycle afterwards. Cycle 1 has burnable poisons and a search on k_{eff} is made. In succeeding cycles, a burnup search is performed. In each cycle, the edge data is generated automatically assuming a checkerboard interior.

A.2 Program Output

A.2.1 Output Description

The output in FLAC is relatively straight-forward and easily understood. There are four different parts to the input - the case data summary, the tabular data edit, the cycle summary and the region summary. A sample output is given in Section A 2.2.

The case data summary merely edits the case input data in a convenient format. The leading lines are printed on the top of each page and give the page number, program name and the case title.

The tabular data is editted next, but it is only given with the first case to reduce redundant printing. The tabular data edits the k_∞ as a

function of burnup and enrichment, the BP worth and axial bias factor as a function of burnup and the isotopic data as a function of burnup and enrichment. The data is editted in the same units as entered.

The cycle summary consists of a separate page summarizing each cycle which was calculated in the case. The first items given are the cycle number and which parameter was to be calculated that cycle (EOL k_{eff} , burnup, feed enrichment). Then the general cycle data is presented. This data includes the cycle burnup, the effective full power hours (EFPH), the elapsed cycle time and corresponding capacity factor, the end of life k_{eff} , the number of burnable poisons and the total core loading.

The batch average data is given next. This data presents the data pertinent to each batch. This includes the enrichment, fuel type and number of assemblies in each batch. The number of assemblies discharged from each batch at the end of the cycle is also given. A negative number here implies that this batch is being used in succeeding cycles, but that more assemblies are being used than are available. Thus, in most cases a negative number signifies an error in the input. The source is just the shuffle input describing the cycle and region from which that batch came. The region ID is the region identifier which is input for each feed region. The region average beginning of cycle and end of cycle burnups are given for each batch in the core.

The depletion data gives the core k_{eff} and region power sharings as a function of burnup. The data given corresponds to the indicated burnup and the power sharings are used to deplete the region over the next time step. The power sharings are normalized to a core average value of 1.000.

The edge data summarizes the region to region coupling that was being used for that cycle's analysis: for each region, the total number of edges

available (four times the number of assemblies), the number of edges used and the number of edges adjacent to the exterior and each region. A mismatch between the number of edges available and the total number of edges used indicates an input error.

The region summary lists each region and gives data pertinent for a fuel cost calculation. Each region is given along with the cycle in which it was a feed and the initial enrichment and fuel type. The region discharge burnup is calculated by averaging the discharge burnups of each discharge batch corresponding to that region. The individual batches are also given. A batch is here referred to as that part of a region which is discharged at the same time. Each batch is indicated by the cycle and region from which it was discharged. Corresponding to each batch, the burnup and elasped time along with the discharge isotopic information is given. The elapsed time includes the cycle time and refueling shutdown time from when the fuel was initially inserted to when it was discharged and includes any intermediate time in the spent fuel pit. The discharge isotopic information is interpolated from the tabular data in the same manner as the k_{∞} given the enrichment, fuel type and discharge burnup.

When the last case has been completed, a message is printed out and the program exits.

A debug option exists which will print out a significant amount of intermediate data. Most of the data is editted using the subroutine BUGOUT for convenience. The BUGOUT edit always includes the array name and subscripts, so determination of the variable corresponding to the debug data is not difficult.

A.2.2 Sample Output

2

Listed on the following pages are sections of the output corresponding to the sample input. Only representative edits are given, but the case titles are given for future comparison purposes.

0. PCM

0. PCM

FLAC TEST CASE - 3 LOOP, 3 REGION CHECKERBOARD CORE - ENRICH SEARCH

NUMBER OF ASSEMBLIES ASSEMBLIES ACROSS CORE COEE HLIGHT AVERAGE CORE LOADING CORE THERMAL POWER	15 0.0 1.00	FT. MTM
PROB. OF ABSORP. (N) ALBEDO MINIMUM ENRICHMENT MAXIMUM ENRICHMENT BURNUP VARIATION PARAM CYCLE BLAS FACTOR	0.300 0.0 4.500 0.0	W/0
BORON WORTH	-10.0	PCM/PPM

DOPPLER DEFECT.....

XENON DEFECT.....

FLAC TEST CASE - 3 LOOP, 3 REGION CHECKFEBOARD CORE - ENRICH SEARCH

K EFF VERSUS DURNUP AND ENKICHMENT

ENGICHMENT		2.600	2.900	3.100
FUEL TYPE		0	0	0
POWER/FIS.		1.600	1.000	1.000
BURNUP BLAS	BP WRTH		K EFF	
$\begin{array}{c} 1.000 & 1.0000 \\ 8.000 & 1.0000 \\ 20.000 & 1.0000 \\ 36.000 & 1.0000 \end{array}$	900.00	1.24408	1.26936	1.28405
	350.00	1.15416	1.18096	1.19698
	25.00	1.02446	1.05247	1.06978
	20.00	0.38846	0.91329	0.92983

101

.

FLAC TEST CASE - 3 LOOP, 3 REGION CHECKERBOARD COBE - ENRICH SEARCH

ISOTOPIC DATA (U) VERSUS BURNUP AND ENRICHMENT

	2.600	2.900	3.100
1.000	0.9984	0.9984	0.9984
8.000	0.9878	0.9878	0.9878
20.000	0.9712	0.9712	0.9712
36.000	0.9524	0.9524	0.5524

FLAC TEST CASE - 3 LOOP, 3 REGION CHECKERBOARD CORE - ENRICH SEARCH

ISOTOPIC DATA (U-35) VELSUS BURNUP AND ENRICHMENT

	2.600	2.900	3.100
1.000	2.4826	2.7819	2.9814
8.000	1.7934	2.0757	2.2653
20.000	1.0048	1.2292	1.3838
36.000	0.4222	0.5558	0.6520

FLAC TEST CASE - 3 LOOP, 3 REGION CHECKERBOARD CORE - ENRICH SEARCH

ISOTOPIC DATA (PU) VERSUS BURNUP AND ENRICHMENT

	2.600	2.900	3.100
1.000	0.5424	0.5320	0.5257
8.000	3.3313	3.3194	3.3105
20.000	5.6175	5.7103	5.7629
36.000	6.6372	6.8515	6.9805

FLAC TEST CASE - 3 LOOP, 3 REGION CHECKERBOARD CORE - ENRICH SEARCH

CYCLE 21

SEARCH PARAMETER. . FLED 1

BURNUP 11.000	GWD/MTM
EF PH	HOURS
CYCLE TIME 13750.0	DAYS
SHUTDOWN LENGTH 42.0	DAYS
CAPACITY FACTOR 80.0	PERCENT
EOL KEFF 1.0011	
NUMBER OF BPS 0.	
LOADING) HTM /

		G	ATCH DAT	A	
		1	2	3	4
	ENRICHHENT	3.211	3.211	3.211	3.211
•	FUEL TYPE	0	0	0	0
	ASSEMBLIES	1	52	52	52
	EISCHARGED	1	51	0	0
	SOURCE-C.R	2102	2103	2104	1
	REGION ID	50	ΕÇ	ΞQ	EQ
	BOC BURNUP	33.036		10.093	0.0
	ROC BURNUP	42.159		22.320	10.093
		Ð	EPLETION	DATA	
BOS BURNU	P K SFF	1	2	3	4
0.0	1.1137	0.801	0.961	1.110	0.933
4.00	0 1.0707	0.833	0.976	1.112	0.916
8.00	0 1.0299	0.862	0.938	1.112	0.903
11.00	1.0011	0.878	0.994	1.108	0.900

·	EDG1.	D AT A					
REGION	AVAIL.	TOTAL	LXTERIOR	1	2	3	4
1	4	4	0	Э	0	4	0
2	208	2 08	0	0	0	180	28
2	208	208	Ō	4	180	0	24
4	203	2 08	60	0	28	24	96

FLAC TEST CASE - 3 LOOP, 3 EEGION CHECKERBOARD CORE - ENKICH SEARCH

REGION SUMMARY

REGION	FEED IN	CYCLE	ENRICH	INDF	ASMBLY	DIS BU	CACTE	ASHBLY	DIS BU
ΕQ	1	21	3,210	0	52	33.212		1 51	

FLAC TEST CASE - 3 LOOP, 3 REGION CHECKERBOARD CORE - 3.10 W/O

CYCLE 21

SEARCH PARAMETEE. BURNUP

BURNUP 10.554	GWD/MTM
EFPH	HOURS
CYCLE TIME 13193. (J DAYS
SHUTDOWN LENGTH 42.1) DAYS
CAPACITY FACTOR 80.1	D PERCENT
LOL KEFF 1.0010)
NUMBER OF BPS 0.	•
LOADING 1.000	D MTM

	В	ATCH DAT	A	•
	1	2	3.	4
ENRICHMENT	3.100	3.100	3.100	3.100
FUEL TYPE	0	Û	0	0
ASSEMBLIES	1	52	52	52
LISCHARGED	. 1	51	()	Ŭ
SOURCE-C,R	2132	2103	2104	1
REGION ID	$\mathbb{P}\mathcal{Q}$	EQ	ЕQ	ΞÇ
BOC BURNUP	31.697	21.337	9.541	0.0
EOC BURNUP	49.565	31.697	21.337	9.541

BOS	BORNUP	DEPLITION DATA				
		K EFF	1	2	3	11. 1. 4 1.
	C.O	1.1107	0.614	0.971	1.118.	0.915
	4.000	1.0671	0.845	0.984	1.117	0.902
	3.000	1.0259	0.873	0.995	1.116	0.392
	10.555	1.0010	0.885	0.999	1.112	0.891

	e DG E	DATA					
REGION	AVAIL.	TOTAL	EXTERIOR	1	2	3	4
1	4	14	0	0	0	4	0
2	208	208	0	0	0	180	28
3	208	208	0	4	180	0	24
4	208	208	60	0	28	24	96

**** FLAC ****

PLAC TEST CASE - 3 LOOP, 3 REGION CHECKERBOARD - AUTO , 3.20 W/O

CYCLE 21

SEARCH PARAMETER..BURNUP

BURNUP		
LFPH	262409.	HOURS
CYCLE TIME	13667.1	DAYS
SHUTDOWN LENGTH		
CAPACITY FACTOR	80.0	PERCENT
EOL KEFF	1.0010	
NUMBER OF BPS	0.	
LOADING	1.000	MTM

Ho

B	A	Ŧ	С	H	Ð	A	T	A

	1	2	.3	-4
ENRICHMENT	3.200	3.200	3.200	3.200
FUEL TYPE	Ŭ	0	0	Û.
ASSEMBLIES	1	52	52	52
DISCHARGED	1	51	0	Û
SOURCE-C,R	2102	2103	2104	1
REGION ID	ΞQ	ΕQ	EQ	EQ
EOC BURNUP	32.339	22.238	10.112	0.0
EOC BURNUP	41.909	32.839	22.238	10.112

		Di	PLETION	DATA	
LOS BURNUP	K EFF	1	2	3	. 4
0.0	1.1131	0.801	0.955	1.107	0.941
4.000	1.0700	0.834	0,972	1.110	0.922
8.000	1.0292	0.862	0.986	1.110	0.907
10.934	1.0010	0.879	0.992	1.107	0.904

ELGL DATA								
REGION	AVAIL.	TOTAL	EXTERIOR	1	2	3	4	
1	4	4	U	Ü	0	4	0	
2	208	208	0	0	14	171	23	
3	208	200	0	4	171	- 4	29	
4	208	208	60	0	23	29	96	

 $\zeta_{\mathcal{J}}$

E

**** FLAC ****

FLAC TEST CASE - 3 LOOP, 3 REGION RING AUTO - 3.20 W/O

CYCLE 21

SEARCH PARAMETER...BURNUP

BURNUP 10.353	GND/HTM
EFPH	HOURS
CYCLE TIME 12940.8	DAYS
SHUTDOWN LENGTH 42.0	DAYS
CAPACITY FACTOR 80.0	PERCENT
EOL KEFF 1.0010) and a second second
NUMBER OF BPS 0.	•
LOADING 1.000	MTM (

	В	ATCH DAT	Α	
	1	2	3	4
ENRICHMENT	3.200	3.200	3.200	3.200
FUEL TYPE	0	0	0	0
ASSEMBLIES	- 1	52	52	52
EISCHARGED	1	51	0	0
SOURCE-C,R	2102	2103	2104	1
REGION ID	EQ	EQ	EQ	EQ
BOC BUENUP	31.142	23.675	10.940	0.0
EOC BURNUP	37.033	31.142	23.675	10.940

			DI	EPLETION	DATA	
BOS	BURNUP	K EFF	1	2	3	4
	0.0	1.1164	0.463	0.626	1.262	1.122
	4.000	1.0676	0.593	0.745	1.224	1.039
	660.3	1.0243	0.708	0.844	1.186	0.975
	10.352	1.0010	0.757	0.883	1.169	0.953

	EDGE	DATA					
REGION	AVAIL.	TOTAL	EXTERIOR	1	2	. 3	- 4
1	4	4	0	0	4	0	0
2	208	2.08	0	4	168	36	0
3	208	208	0	0	36	120	52
4	208	208	60	0	0	52	96

**** FLAC ****

3 LOOP CORE - CYCLES 1-3 2.1/2.6/3.1 CY 1, 3.2 FEED - 3 REGION

CYCLE 1

SEARCH PARAMETER..EOL KEFF

BURNUP	GWD/NTM
DFPH	HOURS
CYCLE TIME	
SHUTDOWN LENGTH 42.0	DAYS
CAPACITY FACTOR 80.0	PERCENT
EOL KEFF 1.0138	. · · · .
NUMBER OF BES 900.	
LOADING	NTM

BATCH DATA 1 2

	1	2	3
ENRICHMENT	2.100	2.600	3.100
FUEL TYPE	0	0	0
BP FRAC.	0.0	0.800	0.200
ASSEMBLIES	53	52	52
DISCHARGED	52	0	• • •
SOURCE-C,R	1	2	3
REGION ID	1	2	3
BOC BURNUP	0.0	Ú. 0	0.0
EOC BURNUP	16.724	16.052	10.681

			Di Di	SPLETION	DATA
BOS	BURNUP	K EFF	1	2	3
	0.0	1.1473	1.215	1.072	0.709
	4.000	1. 1126	1.161	1. 108	0.728
	8.000	1.0768	1.121	1.128	0.749
	12.000	1.0377	1.094	1.129	0.775
	14.500	1.0138	1.080	1.128	0.790

	EDGE	DATA				
REGION	AVAIL.	TOTAL	EXTERIOR	1	2	3
1	212	212	0	18	171	23
2	208	208	0	171	8	29
3	208	208	60	23	29	96

**** FLAC ****

3 LOOP CORE - CYCLES 1-8 2.1/2.6/3.1 CY 1, 3.2 FEED - 3 REGION

CYCLE 2

SEARCH PARAMETER. BURNUP

BURNUP 10.	378 GWD/MTM
EFPH	76. HOURS
CYCLE TIME 1297	2.7 DAYS
SHUTDOWN LENGTH 4	2.0 DAYS
CAPACITY FACTOR 8	0.0 PERCENT
LOL KEFF 1.0	010
NUMBER OF BPS	0.
LOADING 1.	000 MTM

	В	ATCH DAT	A	
	1	2	· 3	4
ENRICHMENT	2.100	2.600	3.100	3.200
FUEL TYPE	0	0	0	0
ASSEMBLIES	1	52	52	52
DISCHARGED	1	51	0	0
SOURCE-C,R	101	102	103	1
REGION ID	1	2	3	4
BOC BURNUP	16.724	16.052	10.681	0.0
EOC BURNUP	26.007	26.165	22.013	9.711

		· · · ·				
BOS	BURNUP	K EFF	1	2	3	4
	Ü.0	1.1106	0.878	0.964	1.089	0.949
	4.000	1.0663	0.898	0.977	1.093	0.932
	8.000	1.0245	0.916	0.987	1.095	0.920
	10.378	1.0010	0.922	0.991	1.093	0.918

	ELGE	DATA					
AEGION	AVAIL.	TOTAL	EXTERIOR	1	.2	 3	4
1	4	4	0	0	0	4	0
2	208	208	0	0	14	171	23
3	208	208	0	4	171	4	29
4	208	208	60	0	23	29	96

**** FLAC ****

3 LOOP CORE - CYCLES 1-8 2.1/2.6/3.1 CY 1, 3.2 FEED - 3 REGION

CYCLE 3

SEARCH PARAMETER..BURNUP

BURNUP 10.823	GWD/MTM
EF PH	HOURS
CYCLE TIME	DAYS
SHUTDOWN LENGTH 42.0	DAYS
CAPACITY FACTOR 80.0	PERCENT
EOL KEFF 1.0010	
NUMBER OF BPS 0.	
LOADING 1.000	MTM

			E	ATCH DAT	Α	1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 -
		•	1	2	3	4
	ENR	ICHAENT	2.600	3.100	3.200	3.200
	FUE	L TYPE	U	0	0	Û
	ASS	EABLIES	1	52	52	52
	EIS	CHARGED	1	51	0	0
	SOU	RCE-C.R	202	203	204	1
	REG.	ION ID	2	3	- 4	5
	BOC	BURNUP	26.165	22.013	9.711	0.0
	EOC	BURNUP	35.232	32.417	21.766	10.044
						•
			D	EPLETION	DATA	
BOS	BJENUP	K BFF	1	2	3.	4
	0.0	1.1124	0.813	0.947	1.113	0.944
	4.000	1.0692	0.842	0.964	1.115	0.925
	8.000	1.0283	0.863	0.978	1.114	0.910
	A 10 M 1					

0.984

1.111

0.907

0.881

10.824

1.0010

	. EDGH	DATA					
REGION	AVAIL.	TOTAL	EXTERIOL	1	2	3	4
1	4	4	6 - 1	0	0	4	0
2	298	208	0	0	14	171	23
3	208	208	Ŭ	4	171	4	29
4	203	208	60	0	23	29	96

**** BLAC ****

3 LOOP CORE - CYCLES 1-8 2.1/2.6/3.1 CY 1, 3.2 FEED - 3 REGION

CYCLE 4

SEARCH PARAMETER...BURNUP

BURNUP	11.107	GWD/MTM
EFPH	266572.	HOURS
CYCLE TIME	13884.0	DAYS
SHUTDOWN LENGTH	42.0	DAYS
CAPACITY FACTOR	80.0	PERCENT
EOL KEFF	1.0010	
NUMBER OF BPS	0.	
LOADING	1.000	MTM

	. B	ATCH DAT	A	
	. 1	2	3	. 4
ENRICHMENT	3.100	3.200	3.200	3.200
FUEL TYPE	υ	0	0	0
ASSEMBLIES	1	52	52	52
DISCHARGED	1	51	0	0
SCURCE-C,R	302	303	304	1
REGION ID	3	- 4	5	6
EOC BURNUP	32.417	21.766	10.044	0.0
EOC BURNUP	41.512	32.593	22.368	10.209
	D	EPLETION	DATA	
BURNUP KEFF	1	2	3	1
0.0 1.1150	0.790	0.961	1.109	0.934
4.000 1.0719	0.822	0.977	1.110	0.910
8.000 1.0310	0.851	0.990	1.110	0.903
11.107 1.0010	0.369	0.996	1.107	0.900

BOS

-	EDGE	D A'L A					
REGION	AVAIL.	TOTAL	EXTERIOR	1	2	3	4
1	4	ં ન	0	υ	0	4	ð
2	208	208	Ŭ	0	14	171	23
. 3	208	2.08	0	4	171	4	29
4	208	203	όO	0	23	29	96

**** FLAC ****

3 LOOP CORE - CYCLES 1-8 2.1/2.6/3.1 CY 1, 3.2 FEED - 3 REGION

REGION	FEED I	N CYCLE	ENRICH	TYPE	ASMBLY	DIS BU	CYCLE	ASMBLY	DIS BU
1	1	1	2.100	0	53	16.899	1 2	52 1	16.724 26.007
2	2	1	2.600	0	52	26.339	2 3	51 1	26.165
3	3	1	3.100	0	52	32.592	3 4	51 1	32.417 41.512
- 4	1	2	3.200	υ	52	32.767	4 5	51 1	32.593 41.636
5	1	3	3.200	0	52	33.050	5 6	51 1	32.876 41.954
6	1	са. <u>4</u> с	3.200	0	52	33.024	6 7	51	32.849 41.916
7	1	5	3.200	0	52	32.995	7 8	51 1	32.820 41.888
8	1	6	3.200	0	52	32.841	8	52	32.841
9	1	7	3.200	0	52	22.232	8	52	22.232
10	1	8	3.200	0	52	10.111	8	52	10.111

REGION SUMMARY

A.3 Programmer's Information

A.3.1 Program Routines

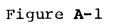
The FLAC program is a relatively simple and straight forward computer program. It consists of 14 subroutines and functions arranged in basically a linear fashion. The flow-chart of the subroutines is depicted in Figure A.1.

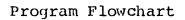
The program is written exclusively in FORTRAN IV and should be operable on either IBM or CDC machines.

The subroutine MAIN is merely a controller for the program. The block DATA routine supplies the default data. All of the input data is entered through the routine READIN. READIN uses a fixed format input with all the formats listed at the end of the subroutine. Automatic generation of data is accomplished in the subroutine PREP. Here the shuffle array and edge data can be generated. Also in this routine is the calculation of the axial leakage correction factor, the fuel isotopic data fit and the number of discharge assemblies from each region in each cycle. The subroutine INEDIT is then called to edit the basic input information. This completes the problem setup. The next group of subroutines actually solve the problem.

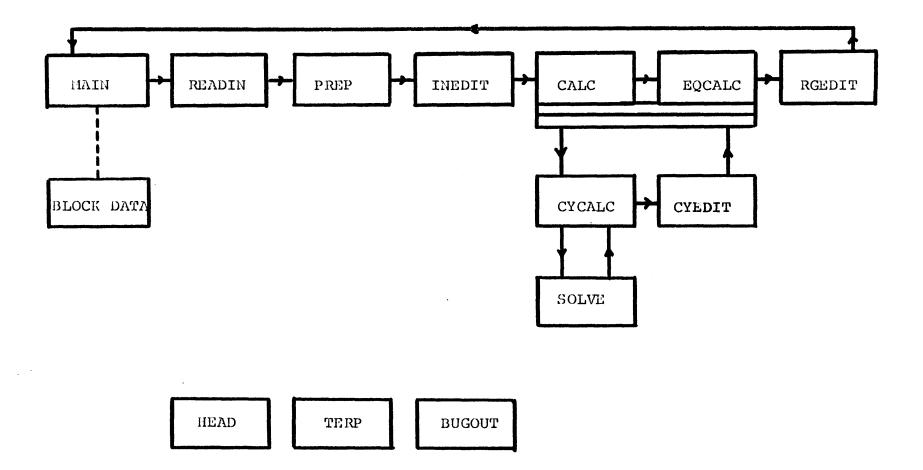
The subroutine CALC sets up the calculation for each cycle except the equilibrium cycle and calls the subroutine CYCALC which does the calculation. For each cycle, CALC sets up the beginning of cycle burnup and the region parameters (enrichment, fuel type, number of assemblies). After the calculation for each cycle has been completed, the subroutine CYEDIT is called to edit the cycle data.

The equilibrium cycle calculation is performed in the subroutine EQCALC. It performs the same function as CALC only for the equilibrium cycle. Also, the search procedure for the desired parameter is carried out





.



in EQCALC since the parameters for the equilibrium cycle are so interlocking. EQCALC also calls CYCALC and CYEDIT.

The actual cycle calculation is done in CYCALC. Here the matrix is set up, the cycle depletion performed and the search for the desired parameter is accomplished. CYCALC is set up to do a straight depletion or a Haling calculation. It calls the function TERP to supply the k_{∞} at the desired enrichment and burnup and calls SOLVE to actually solve the eigenvalue problem.

The subroutine CYEDIT edits the cycle data. The only calculations performed are the calculations of capacity factor or cycle time and the effective full power hours for editting purposes.

After the calculation has been performed for each cycle, the routine RGEDIT is called to generate the region edit. In this routine the fuel cost data set is generated, so coupling with a fuel cost code would be initiated in this subroutine.

Two output routines are called throughout the program. One is HEAD which puts a page heading with the case title at the top of each page. The other routine is BUGOUT which is used extensively for debug printing. BUGOUT only is executed if the debug flag is on. It prints out real or integer variables by specifying the array name, appropriate subscripts and the data.

A.3.2 Program Variables

All of the important variables are in labeled common blocks. The same variable names are used in every subroutine where data is passed through common. All the common blocks and variables in them are listed in Table A.2 along with a description of the variable. Standard FORTRAN

typing nomenclature is used except where noted. All the real variables are single precision except for the variables in the subroutine SOLVE which are double precision. The three real variables transferred to SOLVE in the call are also double precision variables.

TABLE A.2

PROGRAM VARIABLES

Variable	Dimension	Description				
Common block /IC	Common block /IO/					
105	-	Input unit (=5)				
106	-	Output unit (=6)				
107	-	Punch unit (=7)				
Common block /BA	ASIC/					
TITLE	(20)	Case title in A4				
NASS	-	Number of assemblies				
ANASS	-	Real equivalent of NASS				
NADIAM	. -	Number of assembly rows				
NALPHA	-	Number of peripheral assemblies				
ULOAD	· _	Total core loading				
HEIGHT	-	Total core height				
BUDIF	-	Discharge burnup variation parameter				
ISO	-	Iosotpic information generation option				
W	-	Probability of absorption in one neighboring assembly for neutron born in an assembly				
ALPHA	-	Albedo				
ELIM	(2)	Minimum and maximum enrichments for a feed enrichment search				
IHAL	-	Haling calculation option				
POWER	-	Total core thermal power				
DEBUG	-	(Logical) Debug option				
Common block /CO	Common block /COST/					
IF COST	-	Not used				

-

.

.

Variable	Dimension	Description				
Common block /LOOP/						
NA	- -	Number of values of ALPHA to be used in the sequence of cases				
NW	-	Number of values of W to be used in the sequence of cases				
DELA	-	Change in ALPHA between different values				
DELW	-	Change in W between different values				
Common block /CO	EF/					
COFIT	(11)	Not used				
TEMP	(2)	Not used				
BWRTH	-	Signifies if a "boron search" is to be performed at each burnup step				
PWRTH	-	Not used				
XEWRTH	-	Not used				
Common block /TA	BLI/					
NBU	-	Number of burnups in the table				
NRICH	-	Number of enrichments in the table				
BURNUP	(20)	Burnups for tabular data in increasing order				
ENRICH	(20)	Enrichments for tabular data in any order				
ITYPE	(20)	Fuel type corresponding to each enrichment				
BIAS	(20)	Axial bias factor as a function of burnup				
BPWRTH	(20)	Worth of a burnable poison rod in one assembly as a function of burnup				
CYBIAS	-	General bias factor				
Common block /TA	BLE2/					
KINF	(20,20)	(Real) k data as a function of burnup for each enrichment				

Variable	Dimension	Description
ISOTOP	(20,20,6)	(Real) Isotopic data as a function of burnup for each enrichment and for different sets of isotopic parameters
POWFIS	(20)	Power (κ) to fission (ν) ratio for each enrichment
Common Block /CY	CL1/	
CIRICH	(10)	Enrichments for each of the first cycle regions
CITYPE	(10)	(Integer) Fuel type for each region in cycle 1
C1ID	(10)	Individual assembly loading for each region in cycle 1
NACL	(10)	Number of assemblies in each region in cycle 1
Common Block /FI	EED/	
NAFEED	(21,3)	The number of feed assemblies in each cycle for each feed number
FTYPE	(21,3)	(Integer) Type of fuel for each cycle and each feed
FLOAD	(21,3)	The assembly loading for each cycle for each feed
FRICH	(21,3)	The feed enrichment for each cycle and each feed
FID	(21,3)	The feed alphanumeric label (A4) for each cycle and each feed
Common Block /C	YDAT/	
ISURCH	(21)	Search parameter for each cycle
ISHUF	(21)	Shuffle generation option for each cycle
BP	(21)	Number of burnable poison rods for each cycle
CYBU	(21)	Cycle burnup for each cycle

Variable	Dimension	Description
EOLK	(21)	End of cycle k eff for each cycle
NFEED	(21)	Number of feed regions in each cycle
NREG	(21)	Number of regions in each cycle
CYLOAD	(21)	Total core loading for each cycle
CAPFAC	(21)	Capacity factor for each cycle
CYTIM	(21)	Elapsed cycle time for each cycle
SDLEN	(21)	Refueling shutdown length for each cycle
IFEDGE	(21)	Edge data generation option for each cycle
Common block /R	GDAT/	
NEDGE	(2,1000)	Edge data packed according to IEDGE. The first number is of the form 100* I + J where I, J are the adjoining regions and the second number is the number of common edges
IEDGE	(21)	Location of last data item in NEDGE for each cycle. Data for Cycle I is between IEDGE (I-1) + 1 and IEDGE (I)
NAREG	(15,21)	Number of assemblies in each region in each cycle
ISHUF	(15,21)	Shuffle parameter for each region in each cycle. It gives the region and cycle number from which those assemblies came from
BPFRAC	(15,21)	Fraction of burnable poisons in each region in each cycle
BOCBU	(15,21)	Beginning of cycle burnup in each region and in each cycle
EOCBU	(15,21)	End of cycle burnup in each region and in each cycle
IDXREG	(15,21)	Index for each region and each cycle specifying the cycle (K) and feed number (J) in that cycle that was the original source of this region. The number is of the form 100 K + J.

.

.

٠

Variable	Dimension	Description
NADIS	(15,21)	The number of assemblies discharged from each region in each cycle
Common Block /C	YDEP/	
BSHARE	(15,10)	The power sharings in each region at the different depletion steps for a particular cycle
BBU	(10)	The core burnups for the depletion steps for the cycle
BKEFF	(10)	The core k _{eff} at each depletion step for the cycle
NBBU	-	The number of depletion steps
Common Block /E	DGE/	
IDGE	(15,20)	Storage for edge data in a full matrix for a single cycle

•

APPENDIX B

PROGRAM LISTING

On the following pages is the listing of the FLAC computer code for IBM computers. Change cards for modification of the program to enable running on a CDC computer are included as comments with CDC in columns 1-3.

.

CDC CDC	PROGRAM FLAC(INPUT,OUTPUT,PUNCH,TAPE5=INPUT,TAPE6=OUTPUT, 1 TAPE7=PUNCH)	MAIN0001 MAIN0002
C		MAIN0003
č	PROGRAM FLAC - FLARE BASED CYCLING PROGRAM	MAIN0004
C	WRITTEN BY CHARLES L. BEARD, JR. SPRING 1978	MAIN0005
C	NUCLEAR ENGINEERING DEPARTMENT, MIT	MAIN0006
С	NOW WITH WESTINGHOUSE NUCLEAR FUEL DIVISION	MAIN0007
С	PARTIAL FUNDING BY YANKEE ATOMIC ELECTRIC CO.	MAIN0008
С		MAIN0009
С		MAINOO 10
	COMMON /TABL1/ NBU, NRICH, BURNUP(20), ENRICH(20), ITYPE(20), BIAS(20),	MAIN0011
	1 BPWRTH (20), CYBIAS	MAIN0012
	COMMON /IO/ I05, I06, I07	MAINO013
	COMMON /TABL2/ KINF(20,20), ISOTOP(20,20,6), POWFIS(20)	MAIN0014
	REAL KINF, ISOTOP	MAINO015
	COMMON /BASIC/ TITLE(20),NASS,ANASS,NADIAM,NALPHA,ULOAD,HEIGHT,	MAINO016
	1 BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG	MAINOO 17
	LOGICAL DEBUG	MA I NOO 18
	COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21),	MAINO019
	1 NFEED (21), NREG (21), CYLOAD (21), CAPFAC (21), CYTIM (21),	MAINO 020
	2 SDLEN (21), IFEDGE (21)	MAIN0021
	COMMON /RGDAT/ NEDGE(2,1000), IEDGE(21), NAREG(15,21), JSHUF(15,21),	MAIN0022
	1 BPFRAC (15,21), BOCBU (15,21), EOCBU (15,21),	MAIN0023
	2 IDXREG(15,21), NADIS(15,21)	MAINO024
	COMMON /CYCL1/ C1RICH(10), C1TYPE(10), C1ID(10), C1LOAD(10), NAC1(10)	MAIN0025
	COMMON /FEED / NAFEED (21,3), FTYPE (21,3), FLOAD (21,3), FRICH (21,3),	MAIN0026
	1 FID (21,3)	MAIN0027
	INTEGER FTYPE, C1TYPE	MAIN0028
	COMMON /COST / IFCOST	MAIN0029
	COMMON /COEF / COFIT(11), TEMP(2), BWRTH, DPWRTH, XEWRTH	MAIN0030
	COMMON /LOOP/ NA, NW, DELA, DELW	MAIN0031
C 1		MAIN0032
1	0 CONTINUE	MAIN0033
	CALL READIN Ww=W	MAIN0034
	MW = W DO 20 I=1, NA	MAIN0035
	DO ZO I-I, NA	MAIN0036

i i

•

•

	W = W W			MAIN0037
	DO 15 $J=1$, NW			MAINOO38
	CALL PREP			MAIN0039
	CALL INEDIT			MAIN0040
	CALL CALC			MAIN0041
	CALL EQCALC			MAIN0042
	CALL RGEDIT			MAIN0043
	W = W + DELW			MAINOO44
15	CONTINUE			MAIN0045
	ALPHA=ALPHA+DELA			MAIN0046
20	CONTINUE		4	MAIN0047
	GO TO 10			MAIN0048
	END			MAIN0049

i i

•

•

•

•

9

BLOCK DATA	BLKD0001
	BLKD0002
COMMON /TABL1/ NBU, NRICH, BURNUP(20), ENRICH(20), ITYPE(20), BIAS(20),	BLKD0003
1 BPWRTH (20), CYBIAS	BLKD0004
COMMON /10/ 105,106,107	BLKD0005
COMMON /TABL2/ KINF (20,20), ISOTOP (20,20,6), POWFIS (20)	BLKD0006
REAL KINF, ISOTOP	BLKD0007
COMMON /BASIC/ TITLE(20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT,	BLKD0008
1 BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG	BLKD0009
LOGICAL DEBUG	BLKD0010
COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21),	BLKD0011
1 NFEED (21), NREG (21), CYLOAD (21), CAPFAC (21), CYTIM (21),	BLKD0012
2 SDLEN (21), IFEDGE (21)	BLKD0013
COMMON /RGDAT/ NEDGE (2,1000), IEDGE (21), NAREG (15,21), JSHUF (15,21),	BLKD0014
1 BPFRAC (15, 21), BOCbU (15, 21), EOCBU (15, 21),	BLKD0015
2 IDXREG (15, 21), NADIS (15, 21)	BLKD0016
COMMON /CYCL1/ C1RICH(10), C1TYPE(10), C1ID(10), C1LOAD(10), NAC1(10)	BLKD0017
COMMON /FEED / NAFEED (21,3), FTYPE (21,3), FLOAD (21,3), FRICH (21,3),	BLKD0018
1 FID(21,3)	BLKD0019
INTEGER FTYPE, C1TYPE	BLKD0020
COMMON /COST / IFCOST	BLKD0021
COMMON /COEF / COFIT (11), TEMP (2), BWRTH, DPWETH, XEWRTH	BLKD0022
	BLKD0023
DATA BURNUP / 0., 4., 8., 12., 16., 20., 24., 28., 32., 36., 40.,	BLKD0024
1	BLKD0025
DATA NBU, NRICH / 13,0 /	BLKD0026
DATA BPWRTH / $20*0.0$ /	BLKD0027
	BLKD0028
DATA ISURCH, ISHUF, NEDGE, IEDGE / 42*0, 2021*0 /	BLKD0029
DATA IFEDGE / 21*0 /	BLKD0029
DATA NAC1/ 10*0 / · ·	BLKD0030 BLKD0031
DATA COFIT, TEMP, BWRTH, DPWRTH, XEWRTH / 16*0. /	
DATA BPFRAC / 315*0.0 /	BLKD0032
DATA ISOTOP / 2400*0.0 /	BLKD0033
DATA 105,106,107 / 5,6,7 /	BLKD0034
END	BLKD0035

С

С

4

	SUBROUTINE HEAD	HEADOOO1
С		H EA D0002
	COMMON /10/ 105,106,107	HEADOOO3
	COMMON /BASIC/ TITLE(20)	HEAD0004
С		H EA D0005
	DATA NP / O /	HEAD0006
С	, , , , , , , , , , , , , , , , , , ,	H EA D0007
	NP=NP+1	HEAD0008
	WRITE (106,101)	HEAD0009
	WRITE (106,102) TITLE, NP	H EADOO10
	RETURN	HEADOO11
С		H EA D0012
101	FORMAT(1H1,48X,14H**** FLAC ****)	HEADOO13
102	PORMAT (1H0, 14X, 20A4, 10X, 4HPAGE, I3 //)	HEADOO14
С		HEADOO15
	END	HEADOO 16
	•	

•

.

.

	SUBROUTINE BUGOUT (NAME, ARRAY, I1, I2, I3, J1, J2, J3, IT)	BUGO0001
С		BUG00002
	COMMON /IO/ IO5,IO6,IO7 ·	BUGO0003
	COMMON /BASIC/ TITLE (20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT,	BUGOOOO4
	1 BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG	BUGO0005
	LOGICAL DEBUG	BUGOOOO6
	DIMENSION ARRAY (I3, J3), NAME (2)	BUGO0007
С		BUGOOOOB
	IF (.NOT.DEBUG) GO TO 100	BUG00009
	IF (I3.GT.1.OR.J3.GT.1) GO TO 10	BUG00010
	IF (IT.EQ.2) GO TO 5	BUG00011
	WRITE (106,101) NAME,ARRAY(1,1)	BUG00012
	GO TO 100	BUG00013
	5 CONTINUE	BUG00014
	WRITE (106,102) NAME, ARRAY (1,1)	BUG00015
	GO TO 100	BUGO0016
	10 CONTINUE	BUG00017
	WRITE (106,103)	BUG00018
	$DO \ 30 \ J=J1, J2$	BUG00019
	DO 25 I=11,I2,8	BUG00020
	II=MINO(I2,I+7)	BUG00021
	IF (IT.EQ.2) GO TO 15	BUG00022
	WRITE $(106, 104)$ NAME, I, J, $(ARRAY(K, J), K=I, II)$	BUGO0023
	GO TO 25	BUG00024
	15 CONTINUE	BUGO0025
	WRITE (106,105) NAME, I, J, (ARRAY (K, J) , $K=I$, II)	BUG00026
	25 CONTINUE	BUG00027
	30 CONTINUE	BUGO0028
	100 RETURN	BUG00029
С		BUGO0030
	101 FORMAT (1H0, 9X, A4, A2, 1H=, I12)	BUG00031
	102 FORMAT (1H0, $9X$, A4, A2, 1H=, E12.5)	BUGO0032
	103 FORMAT(1X)	BUG00033
	104 FOR MAT $(10X, A4, A2, 1H(, I3, 1H, I3, 2H) = , 8I12)$	BUGOOO34
	105 FORMAT (10X, A4, A2, 1H (, I3, 1H, , I3, 2H) = $,8 E12.5$)	BUG00035
С		BUG00036

BUG00037

	TERPOOO1
FUNCTION TERP (B, E, IT, XX, IOPT, ND, DPN)	TERP0002
RUNCHTON TERD INTERDOLATION RUNCTION FOR ARRAYS BASED ON	# 29 D () () 3
REPAID AND ENSTER DOUBLE INTERPOLATION HARARYS DESERVED ON	TERP0004
BURNUF AND ENRICH. DUUDEE INTERCOMMINUM UNING DIOINT ILLU BUDNED RITH IS ONR DOINT BRIOW AND TWO AROVE IF POSSIBLE.	TERP0005
	TERP0006
	TERP0008
DEGENERALE TO LINEAR IT EXTRAPOLATION REQUIRED.	TERP0009
ИЛДЛАТВОС ТИ САТТ СОХОДИДАТО ХОД	TERPOOIO
	TERP0011
	TERPOO 12
	TERPOOIS TERPOOIS
	TERPOO15
, BURNUP, SECOND INDER IS ENKLED TODE CONTON TE -1 MHEN INCO CALLED NICH & NEW ADDAY -	
LOPI OPILON, IF -1, INEN JUSI CALLED WITH A NEW ARGAI -	
	TERPOO18
•	TERPOOIO
	TERPOOLO
BPK BP WOITH PER ROD AT BURNUP BU	TERP0020
ΟΟΜΜΟΝ (ΠΑΒΙΊ (ΝΟΠ ΝΟΤΟΠ ΟΠΟΝΠΟ(20) ΕΝΡΤΟΠ(20) ΤΟΥΡΕ(20) ΕΤΧΟ(20)	TERP0022
	TERPOO23
	TERP0025 TERP0024
	T ERP0024
	TERPOO26
	TERP0028
	TERP0029
	TERP0030
	TERP0031
	TERPO032
	TERP0033
	TERPOO34
	TERP0035
COMMON /CYCL1/ CIRICH (10), CITYPE (10), CIID (10), CILOAD (10), NAC1 (10)	TERP0036
	<pre>FUNCTION TERP (B, E, IT, XX, IOPT, Kb , BPK) FUNCTION TERP INTERPOLATION PUNCTION FOR ARRAYS BASED ON BUBNUP AND ENRICH. DOUBLE INTERPOLATION UNING 3 POINT PIT. BUKNUP PIT IS ONE POINT BELOW AND TWO ABOVE IF POSSIBLE. ENNICH FIR IS ONE POINT BELOW, ONE POINT ABOVE AND NEXT CLOSEST POINT IF THIS IS POSSIBLE. ENRICH FIT WILL AUTO- DEGENERATE TO LINEAR IF EXTRAPOLATION REQUIRED. VARIALBES IN CALL STATEMENT ARE - B EURNOP AT WHICH DATA IS WANTED E ENRICHMENT AT WHICH DATA IS WANTED IT TYPE OF FUEL - INTERPOLATION USES ONLY THAT FUEL TYPE XX(20,20) ARRAY TO BE INTERPOLATED - FIRST INDEX IS BURNUP, SECOND INDEX IS ENRICH IOPT OPTION, IF =1, THEN JUST CALLED WITH A NEW ARRAY - POINT AND INTERPOLATE ONLY ON ENWICHMENT. KB (KEAL) K BIAS VALUE BEK BP WOTTH PER KOD AT BURNUP BU COMMON /IABL2/ KINF(20,20), ISOTOP (20,20,6), POWFIS(20) REAL KINF, ISOTOP COMMON /CYDAT/ ISURCH(21), SIGTOP (20,20,6), POWFIS(20) REAL KINF, ISOTOP COMMON /CYDAT/ ISURCH(21), SIGTOP (21), CYBU(21), EOLK(21), 1</pre>

.

.

	COMMON /FEED / NAFEED (21,3), FTYPE (21,3), FLOAD (21,3), FRICH (21,3),	TERP0037
	1 FID (21,3)	TERP0038
	INTEGER FTYPE, C1TYPE	TERPOO39 TERPOO40
	COMMON /COST / IFCOST	TERP0040 TERP0041
	COMMON /COEF / COFIT (11), TEMP (2), BWRTH, DPWRTH, XEWRTH	
	REAL KB	TERP0042
	DIMENSION XX(20,20), EMIN(3), INDEX(3)	TERP0043
	DATA IERR / 0 /	TERPOO44
С		TERP0045
	IF (IOPT.EQ.1) GO TO 65	TERP0046
	IF (IOPT.EQ.2) GO TO 12	TERPOO47
	I=NBU	TERP0048
		TERP0049
	DO 10 $K=2$, NBU	TERP0050
	IF $(B.GT.BURNUP(K-1).AND.B.LE.BURNUP(K))$ I=K	TERP0051
	10 CONTINUE	TERP0052
	IF (B.LE.BURNUP(1)) $I=2$	TERP0053
	I = MINO(I, NBU-1)	TERP0054
	11=1-1	TERP0055
	12=I	TERP0056
	I3=I+1	TERP0057
	B1 = (BURNUP(I2) - B) * (BURNUP(I3) - B) / (BURNUP(I2) - BURNUP(I1)) /	TERP0058
	$1 \qquad (BUKNUP(13) - BURNUP(11))$	TERP0059
	B2= (BURNUP(I1)-B) * (BURNUP(I3)-B) / (BURNUP(I1)-BURNUP(I2))/	TERP0060
	2 (BURNUP(I3)-BURNUP(I2))	TERP0061
	B3=(BURNUP(I1)-B)*(BURNUP(I2)-B) / (BURNUP(I1)-BURNUP(I3))/	TERP0062
	3 (BURNUP (I2) - BURNUP (I3))	TERP 006 3
	KB=B1*BIAS (I1) +B2*BIAS (I2) +B3*BIAS (I3)	TERP0064
	BPK=1_E-5*(B1*BPWRTH(I1)+B2*BPWRTH(I2)+B3*BPWRTH(I3))	TERP0065
	KB=KB*CYBIAS	TERP0066
С		TERP0067
	12 CONTINUE	TERP0068
	EMIN(2) = 100.	TERP0069
	EMIN(3) = 100.	TERP0070
	EMIN(1) = -100.	TERPO071
	INDEX(1) = 0	TERP0072

c

	INDEX(2)=0	TERP0073
	INDEX(3) = 0	TERP0074
	N = 0	TERP0075
	DO 30 K=1, NRICH	TERP0076
	IF (ITYPE(K).NE.IT) GO TO 30	TERP0077
	ERR=ENRICH (K) -E	TERP0078
	IF (ERR.LT.0.0) GO TO 15	TERP0079
	IF (ERR.EQ.0.0) GO TO 20	TERP0080
	IF (EMIN(2).LT.ERR) GO TO20	TERP0081
	IF (EMIN (2).EQ.ERR) GO TO 30	TERP0082
	ER = ABS(EMIN(2))	TERP0083
	KK=INDEX(2)	TERP0084
	EMIN(2) = ERB	TERP0085
	INDEX(2) = K	TERP0086
	GO TO 25	TERPOO87
15	CONTINUE	TERP0088
	IF (EMIN(1).GT.ERR) GO TO 20	TERP0089
	IF (EMIN(1).EQ.ERR) GO TO 30	TERP0090
	ER = ABS(EMIN(1))	TERP0091
	KK=INDEX(1)	TERP0092
	EMIN(1) = ERR	TERP0093
	INDEX(1) = K	TERP0094
	GO TO 25	TERP0095
20	CONTINUE	TERP0096
	ER=ABS (ERR)	TERP0097
	K K = K	TERP0098
25	IF (ER.GE.EMIN(3)) GO TO 30	TERP0099
	EMIN(3) = ER	TERP0 100
	INDEX(3) = KK	TERP0101
30	CONTINUE	TERP0102
	$\mathbf{N} = \mathbf{O}$	TERP0103
	DU 35 K=1,3	TERP0104
	J=INDEX(K)	TERP0105
	IF $(ABS(EMIN(K)) \cdot EQ. 100.) EMIN(K) = 100.$	TERP0106
	IF $(EMIN(K) \cdot NE \cdot 100 \cdot) EMIN(K) = ENRICH(J)$	TERP0107
	IF $(EMIN(K) - NE - 100) = N + 1$	TERP0108

35	CONTINUE		TEBP0109
	DO 40 K=1,2		TERP0110
	K 1= K + 1		TERP0111
	DO 40 J=K1,3	r	TERP0112
	1F (EMIN(J).GT.EMIN(K)) GO TO 40		TERP0113
	EE = EMIN(K)	, i i i i i i i i i i i i i i i i i i i	TERP0114
	EMIN(K) = EMIN(J)		TERP0115
	EMIN(J) = EE	1	TERP0116
	J1=INDEX(K)		TERP0117
	INDEX (K) = INDEX (J)	•	TERPO118
	INDEX $(J) = J1$		TERP0119
40	CONTINUE	1	TERP0120
	IF (N.EQ.0) WRITE (106,102) IT		TERP0121
	IF (N.EQ.O) IERR=IERR+1	·	TERP0122
	IF (N.EQ.0) GO TO 70	1	TERP0123
	GO TO (45,50,60), N		TERP0124
45	CONTINUE		TERP0125
	J1=INDEX(1)		TERP0126
	$J_2=INDEX(1)$		TERP0127
	J3=INDEX (1)		TERP0128
	E 1= 1. 0		TERP0129
	$E_{2}=0.0$	ſ	TERP0130
	£3=0.0	1	TERP0131
	GO TO 65	,	TERP0132
50	CONTINUE	•	TERP0133
	J = INDEX (1)		TERP0134
	J2=INDEX(2)		TERP0135
	J J = I N D E X (2)	·	TERP0136
	$E_{3=0.0}$		TERP0137
	E2 = (ENRICH(J1) - E) / (ENRICH(J1) - ENRICH(J2))	4	TERP0138
	E 1= 1.0-E2		TERP0139
	GO TO 65	•	TERP0140
60	CONTINUE	1	TERP0141
	J1=INDEX(1)	·	TERP0142
	J2=INDEX(2)		TERP0143
	$JJ = IND \mathbb{Z}X(3)$	•	TERP0144

	E1 = (ENRICH (J2) - E) * (ENRICH (J3) - E) / (ENRICH (J2) - ENRICH (J1)) / (ENRICH (J2) - ENRICH (J1)) / (ENRICH (J2) - ENRICH (J2)) / (ENRICH (TERP0145
	1 (ENRICH $(J3)$ - ENRICH $(J1)$)	TERP0146
·	E2 = (ENRICH(J1) - E) * (ENRICH(J3) - E) / (ENRICH(J1) - ENRICH(J2)) /	TERP0147
	2 (ENRICH (J3) - ENRICH (J2))	TERPO 148
	E3=(ENRICH(J1)-E)*(ENRICH(J2)-E)/ (ENRICH(J1)-ENRICH(J3)) /	TERP0149
	$3 \qquad (ENRICH(J2) - ENRICH(J3))$	TERP0150
	65 CONTINUE	TERP0151
C		TERP0152
	IF (IOPT.NE.2) GO TO 66	TERP0153
	TERP = E1 * XX (J1,1) + E2 * XX (J2,1) + E3 * XX (J3,1)	TERP0154
	GO TO 70	TERP0155
С		TERP0156
	66 CONTINUE	TERP0157
	X1= B1*XX(I1,J1) + B2*XX(I2,J1) + B3*XX(I3,J1)	TERP0158
	X2 = B1 * XX (I1, J2) + B2 * XX (I2, J2) + B3 * XX (I3, J2)	TERP0159
	X3 = B1 * XX (I1, J3) + B2 * XX (I2, J3) + B3 * XX (I3, J3)	T ERP0160
	TERP=E1*X1+E2*X2+E3*X3	TERP0161
	70 CONTINUE	TERP0162
	IF (DEBUG) WRITE (IO6,101) B,E,IT,TERP	TERP0163
	IF (IERR.GT.10) CALL EXIT	TERP0164
	RETURN	TERP0165
С		TERP0166
	101 FORMAT (10X, 11HTERP AT BU= ,F9.3, 3H E=,F6.2,7H, TYPE=,I2,5H, IS ,	TERP0167
	1 E12.5)	TERP0168
	102 FORMAT (24HOERROR IN TERP FOR TYPE , 13)	TERP0169
C	· · · · · · · · · · · · · · · · · · ·	TERP0170
	END	TERP0171
		THEOTH

· .

.

.

~	SUBROUTINE READIN	READOOO1 READOOO2
C	SUBROUTINE READIN - READS IN ALL INPUT DATA	READ0003
C	SUBROUTINE READIN - READS IN ALL IMPOI DAIR	READ0004
C	INPUT DATA	READ0005
C C	INPUT DATA	READ0006
C		READ0007
C C	THE FOLLOWING INPUT FORMAT IS USED. FOR MULTIPLE CASES STACK THE	READ0008
c	DECKS. THE PROGRAM WILL CHECK FOR END OF DATA.	READ0009
C	DECKS. THE FROGRAM WILL CHECK FOR MAD OF DATA.	READ0010
C	INPUT PARAMETERS IN ORDER ARE:	READO011
C	INFOI FARANLILAS IN ORDER ARL.	READO012
c	COLS. PARAM. TYPE SUBPT UNITS DESCRIPTION	READ0013
c	COLD. IARAM. III BODII GALLS BABGALLICA	READOO14
C		READ0015
c	**** TITLE CARD (20A4)	READOO 16
ĉ		READ0017
Č	1-80 TITLE A (20) FULL CARD OF TITLE DATA FOR CASE	R EADOO 18
C		READO019
с с	**** CASE DATA CARD A (2413)	READOO20
С		READ0021
C	1- 3 NASS I NUMBER OF ASSEMBLIES IN WHOLE CORE	READ0022
С		READ0023
С	4- 6 NADIAM I NUMBER OF ASSEMBLIES ACROSS CORE	READ0024
С		READ0025
С	7-9 NBU I NUMBER OF BURNUPS IN INTERPOLATION	READ0026
С	TABLE. IF NOT ENTERED, USE DEFAULT	READ0026 READ0027 READ0028
С С С С	ON FREATOOD CADE DORAGED.	MLADV020
C	IF =-1, THEN TABLES INPUT AS	READ0029
С	COEFFICIENTS OF DEFINED FUNCTIONS	
С	AS IN SIMULATE.	READ0031
С		READOO32
C	10-12 NRICH I NUMBER OF K INF. SETS TO BE READ	READ0033
С	IN. IF =0, USE DEFAULT OR PREVIOUS	READOO34
C	CASE DATA, IF >0, REPLACE PREVIOUS	R EA D0035
C	CASE DATA, IF >0, REPLACE PREVIOUS SET WITH NEW SET. IF <0, ADD THIS	READOO36

С					MANY NEW SETS TO PREVIOUS SETS.	READ0037 READ0038
C	13-15 ISO	I			IF =1, READ IN ISOTOPIC DATA ALONG	READOO39
C C	13-12 120	Ŧ			WITH K INF. DATA.	READ0040
C					IF =-1, GENERATE ISOTOPIC DATA FROM	READOO41
c					MUDDLE FIT.	R EA DOO42
C C						READOO43
C	16-18 IFCOST	т			IF =1, PREPARE DATA SET FOR A FUEL	READO044
c	10 10 ILCODI	1			CYCLE COST CALCULATION.	READ0045
c						READ0046
c	19-21 INFIT	I			IF = 1, READ IN FIT DATA.	READ0047
C		-				READ0048
č	22-24 IHAL	I			IF = 1, A HALING CALCULATION WILL BE	REA D0049
c		-			DONE FOR EACH CYCLE.	READ0050
c						READ0051
č	25-27 DEBUG	L			IF =1, DEBUG PRINT OPTION	READ0052
С					•	R EA D0053
С	28-30 NW	I			NUMBER OF DIFFERENT VALUES OF W TO	R EA DO 054
C					USE. IF NOT ENTERED, SET TO 1.	READ0055
С						READ0056
С	31-33 NA	Ι			NUMBER OF DIFFERENT VALUES OF ALPHA	READ0057
С					TO USE. IF NOT ENTERED, SET TO 1.	REA D0058
C C C						READ0059
С						REA D0060
	**** CASE D	АТА С	CARD B	(12F 6)	.2)	READO061
C					·	R EA D0062
C	1-6 W	R			PROBABILITY THAT A NEUTRON BORN IN	R EADOO63
С С					AN ASSEMBLY WILL BE ABSORBED IN AN	READ0064
C					ADJACENT ASSEMBLY. DEFAULT = 0.115.	READ0065
С						READ0066
С	7-12 ALPHA	R			ALBEDO ON EXTERIOR EDGE OF CORE.	R EA D0067
С					DEFAULT = 0.30.	READ0068
С						REA D006 9
С	13-18 ELIM	К	(1)	W/0	MINIMUM FEED ENRICHMENT IN SEARCH	READ0070
С						R EA D0071
С	19-24 ELIM	R	(2)	W/0	MAXIMUM FEED ENRICHMENT IN SEARCH	R EAD0072

					IF NOT INPUT, SET TO 4.5	READOO
					IL NOT INFOLD DEL LO 4.5	READOO
25-30	BUDIF	R		GWDZMTM	BURNUP DIFFERENCE FOR SPLIT REGION	READOO
20 20	55521	-		0.0/ 111	BURNUP OF SPLIT REGION IS GIVEN BY	R EA DOO
					$BU = AVG_BU - BUDIF*(1-N KEPT)/N)$	READOO
						READOO
31-36	CYBIAS	R			K EFF BIAS FACTOR - DIVIDES INTO K	READOO
						READOOR
37-42	HEIGHT	R		FEET	CORE HEIGHT FOR GENERATION OF AXIAL	READOO
					BIAS FACTOR AS A FCN OF BURNUP.	READOOR
						READOOL
43-48	ULOAD	R		MTM	TOTAL CORE AVERAGE LOADING.	READOOR
						READOOR
49-54	POWER	R		MW T	TOTAL CORE POWER.	READOO
						READOO
55-60	DELW	R			INCREMENTAL CHANGE IN W TO BE USED	R EADOO
					TO GENERATE NW-1 VALUES OF W AFTER	READOO
					THE BASE INPUT VALUE.	READOO
						READOOS
61-66	DELA	R			INCREMENTAL CHANGE IN ALPHA TO BE	R EA DOO
					USED TO GENERATE NA-1 VALUES OF	R ZADOO
					ALPHA AFTER THE BASE INPUT VALUE.	READOO
						READOOS
						REA DOOS
****	BURNUP	DATA	CARDS	5 (12 F6.	.2) (ONLY IF NBU>O)	READOOS
						READOOS
	BURNUP	R	(I)	GWD/MTM	BURNUP ARRAY IN ORDER OF INCREASING	READOO
					BURNUP. 12 ITEMS PER CARD, UP TO 20	READO 10
					ENTRIES READ IN. ONLY READ IN IF	READ01
					NBU>O, THEN NBU ENTRIES READ	READO 10
						READ010
						READO 10
****	AXIAL	BIAS	CARDS	(6E12.5	(ONLY IF NBU>0, HEIGHT=0)	R EADO1(
						R EADO 10
	BIAS	R	(1)		AXIAL BIAS FACTOR AS A FUNCTION BU,	READO1(
					ENTERED ONLY IF NBU>O AND HEIGHT=O.	READ010

					NBU ENTRIES CORRESPONDING TO BURNUP(I), 6 ENTRIES PER CARD.	READ0109 READ011(
						READO11
						READ0112
****	BP WOR	RTH C	ARDS	(6E12.	5) (ONLY IF NBU>O)	READO113
						R EAD0114
	BP	R	(I)	PCM	WORTH OF ONE BP ROD/ ASSEMBLY AS A	READO115
					FUNCTION OF BURNUP (I). ENTERED ONLY	READ0116
					IF NBU>0. NBU ENTRIES, 6 PER CARD.	READ0117
						READO118
						READ0119
****	BP COE	EFFIC	IENT CA	RD (6E1	2.5) (ONLY IF NBU<0)	READ0120
						R EA D0121
	В	R	(I)		COEFFICIENTS OF FUNCTION FOR WORTH	READO 122
						READ0123
						R EADO 124
					B(4) *BU**3 + B(5) *BU**5)	READ0125
					WHERE BU IS THE BURNUP IN GWD/MTU.	READ0126
						READO 127
* * * *	ENDIC		CARDO	(1)777		R EA D0128
* * * *	ENRICE	MENT	CARDS	(12F6-	2) (ONLY IF NRICH NE O)	READO129
	END T CC	1 Ta	(T)			READ0130
	ENRICE	1 R	(I)	W/0	ENRICHMENT CORRESPONDING TO EACH	READO131
					DATA SET. W/O U-235 OR W/O PU.	R EA DO132
					NRICH VALUES ENTERED, 12 PER CARD.	READO133
						READO134
****	FUEL 1	VDF	CADDC	(1216)	·	READ0135
	FUEL 1	. 1 - 6	CALDS	(1210)	(ONLY IF NHICH NE O)	READ0136
	ITYPE	I	(1)		FUEL TYPE CORRESPONDING TO EACH	READ0137 READ0138
	* 1 1 C D	.	(1)		DATA SET. NRICH VALUES ENTERED 12	READO138
					PER CARD.	READOIS READOI40
						R EADO140
						R EADO 142 R EADO 142
* * * *	POWER	FISS	ፐርነክ ጽሏሞ	TO CARD	S (12F6.0) (ONLY IF NRICH NE O)	READ0143

~

C C C C		POWFIS R (I) MEV/N POWER TO FISSION SOURCE RATIO FOR EACH FUEL TYPE. (KAPPA/NU) NRICH VALUES, 12 PER CARD.	READO147 READO148
C C			READO149
C C	***	K 1NF CARDS (6E12.5) (ONLY IF NRICH NE 0, NBU GE 0)	R EA D0150 R EAD0 151
C C	****	A THE CARDS (OLIZOS) (ONLY IF WATCH WE U, WOU GE U)	READO151 READ0152
C		KINF R (I,J) K INF CORRESPONDING TO BURNUP(I)	READ0152 READ0153
C C			
C C		EACH SET, 6 VALUES PER CARD, NRICH	R EA DO 154 R EA DO 155
C		SETS.	READOISS READO156
c		2772•	READO150
C			READO 158
č	****	KINF COEFFICIENT CARDS (6E12.5) (ONLY IF NRICH NE 0, NBU<0)	READ0150
c		ATH CONTROLMAL CARDS (CHIERS) (CHAR IT WATCH HE OF HECO	READO 160
ĉ		B R (I) COEFFICIENTS OF FUNCTION FOR K INF.	READO161
C			READ0162
С		ENTERED FOR EACH ENRICHMENT.	READO163
С			READ0164
С			READ0165
С		Z = 1.0 - B(2) - B(3) * (1.0 + B(4) * BU)	
С		$Z_2 = 1.0 - B(5) * BU + B(6) * BU * 2 +$	READ0167
С		B(7) *BU**3 + B(8) *BU**4	READ0168
С		WHERE BU IS BURNUP IN GWD/MTU.	READ0169
С			READ0170
С		ENTERED, SO 2 CARDS/SET ARE NEEDED.	READO 171
С		- <i>,</i> ,	R EA D0172
C			R EADO 173
С	****	ISOTOPIC DATA CARDS (6E12.5) (ONLY IF ISO=1, NRICH NE 0)	READ0174
С			DRADO175
C		ISOTOP R (1, J, 1) KG/NTM URANIUM CONTENT AS A FUNCTION OF BURNUP(I). NBU ENTRIES, 6 ENTRIES PER CARD FOR ENRICH(J).	READ0 176
С		BURNUP(I). NBU ENTRIES, 6 ENTRIES	REA D0177
С		PER CARD FOR ENRICH (J).	READO 178
С			R EA D0179
С		ISOTOP & $(I, J, 2)$ KG/MTM U-235 CONTENT AS A FUNCTION OF	READO 180

C C						BURNUP(I). NBU ENTRIES, 6 ENTRIES PER CARD FOR ENRICH(J).	READO181 READO182
C							READO 183
č		ISOTOP	R	(I.J.)	3) KG /M TM	TOTAL PU CONTENT AS A FUNCTION OF BURNUP(I). NBU ENTRIES, 6 ENTRIES	READO184
č						BURNUP(I). NBU ENTRIES, 6 ENTRIES	READO 185
c						PEH CARD FOR ENRICH (J).	READ0186
č						•••	R E A D O 187
c		ISOTOP	R	(I, J,	4)	FISSILE/(TOTAL PU) RATIO AS A	READO188
c				••••	•	FUNCTION OF BURNUP(I) FOR ENRICH(J).	READO 189
Ċ							READ0190
c		NOTE	***	THE	ORDER OF	CARDS ARE ALL BURNUPS IN SEQUENCE	READ0191
С) FORM A SET. THEN EACH SET FOR	READ0192
C C				DIFF	ERENT ISC	DTOPIC INFORMATION (4 SETS) FOR	READ0193
С				ENRIC	CH(J) TO	FORM A LARGER SET WHICH IS THEN REPEATED	READO 194
С				FOR 1	EACH ENRI	LCHMENT. IF NOT ENTERED, DEFAULT	R EA D0195
С				DATA	WILL BE	USED FROM FIT. DATA IS ONLY USED	READO196
С				FOR	COST DATI	A SET PREPARATION.	READ0197
С							READ0198
C						(5F6.2/ (6E12.5)/) (ONLY IF INFIT =1)	R EA D0199
С	****	COEFFIC	I EN T	FIT	DATA SET	(5F6.2/ (6E12.5)/)	READO200
С						(ONLY IF INFIT =1)	READ0201
С С							READ0202
C	1- b	XEWRTH	R		PCM		READ0203
С							READO204
С	7-12	DPWRTH	R		PC M		READ0205
С						AT BOL.	R EA D0206
C							READO207
С	13-18	BWRTH	R		PCM/PPM	BORON WORTH AT BOL. HZP TEMPERATURE.	R EA DO208
C							READ0209
С	19-24	TEMP	R	(1)	DEG F	HZP TEMPERATURE.	READ0210
С							READ0211
C	25-30	TEMP	R	(2)	DEG F	AVERAGE CORE HFP TEMPERATURE.	READ0212
С							READ0213
C		COFIT	R	(I)		MODERATOR TEMPERATURE FIT PARAMETERS	READO214
C						UP TO 10 ALLOWED, 6 PER CARD.	R EA D0215
С							READO216

С							READ0217
С	* * * *	CYCLE	DATA *	*** A	LL OF T	HE REMAINING DATA IS INPUT FOR EACH	READ0218
С					CYCLE,	THEN FOR THE NEXT CYCLE, ETC. THE	READO219
C					CYCLES	THEN FOR THE NEXT CYCLE, ETC. THE SHOULD BE IN INCREASING ORDER.	READ0220
С					ONE MUS	ST START WITH CYCLE 1. BUT BOC	READO221
С					BURNUP	S MAY BE ENTERED. NO CYCLE MAY BE	READ0222
С С С С					ELIMIN	S MAY BE ENTERED. NO CYCLE MAY BE ATED FROM THE SEQUENCE, EXCEPT AT D. THUS ANY NUMBER OF CYCLES UP TO BE RUN AND THEY MUST BE CONSECUTIVE. 21 IS AN EQUILIBRIUM CYCLE AND IS TE FROM THE REST. THE CYCLE DATA E FOLLOWED BY A BLANK CARD AFTER THE YCLE INPUT.	READ0223
С					THE ENI	D. THUS ANY NUMBER OF CYCLES UP TO	R EA D0224
С					20 MAY	BE RUN AND THEY MUST BE CONSECUTIVE.	READ0225
С					CYCLE 2	21 IS AN EQUILIBRIUM CYCLE AND IS	READ0226
С					SEPARA	TE FROM THE REST. THE CYCLE DATA	READ0227
С					MUST BI	E FOLLOWED BY A BLANK CARD AFTER THE	READO228
С					LAST C	YCLE INPUT.	READO229
С							R EA DO230
С							R EADO 231 READO 232
С	***	CYCLE	OPTION	CARD	(8I 3 , (6F6.0)	READ0232
С							86100333
С	1-3	IS	1			CYCLE NUMBER 1-20 OR 21 FOR EQUIL.	READ0234
C						CYCLE.	READ0235
С							READ0236
С	4- 6	ISURCH	1	(IS)		SEARCH PARAMETER FOR CYCLE IS.	READ0237
С						=1 SEARCH ON CYCLE BURNUP	READ0238
С С С С						 =1 SEARCH ON CYCLE BURNUP =2 SEARCH ON NUMBER OF BPS =3 DETERMINE EOL K EFF. =10+J SEARCH ON ENRICHMENT FOR 	R EA D0239
C						=3 DETERMINE EOL K EFF.	R E ADO 2 40
С						=10+J SEARCH ON ENRICHMENT FOR	READ0241
C						FEED J IN CYCLE IS	READ0242
С							R EADO 243 R EADO 244 R EADO 245
С	7- 9	ISHUF	I	(IS)		SHUFFLE GENERATION OPTION	R EA D 0 2 4 4
С						=0 INPUT SHUFFLE FOR THIS CYCLE	READ0245
C C						=-1 USE SHUFFLE FROM LAST CASE FOR	READO246
С						THIS CYCLE.	READO247
С						>0 GENERATE SHUFFLE DATA AUTOMATICALLY	R EA DO248
С						=1 ASSUME PERITHERAL FEED AND CENTER	READO249
С						ASSEMBLY FROM PREVIOUS CYCLE. =2 ASSUME PERIPHERAL FEED AND CENTER	READ0250
C						=2 ASSUME PERIPHERAL FEED AND CENTER	READ0251
С						ASSEMBLY FROM CYCLE 1.	R EA D0252

C C	10-12	NDEC	т	(10)		NUMBER OF DECTONS IN CYCLE (216)	READ0253 READ0254
C	10-12	NREG	I	(IS)		NUMBER OF REGIONS IN CYCLE (<16)	READ0254
C	13-15	NFEED	I	(15)		NUMBER OF FEED REGIONS IN CYCLE.	READO256
C	15 15		-	(15)		MAY BE $0, 1, 2, \text{ OR } 3.$	READO257
c							READ0258
c	16-18	TROC	I			=1 IF BOC REGION BURNUPS ARE TO BE	READ0259
Ċ		1000	-			ENTERED FOR THIS CYCLE.	READ0260
C						=0 IF BOC REGION BURNUPS ARE TO BE	READ0261
č						CALCULATED USING SHUFFLE DATA AND	READ0262
C						EOC REGION BURNUPS FROM PREVIOUS	READO263
c						CYCLES.	READ0264
ĉ						=-1 IF PREVIOUS CASE BOC REGION	READ0265
С						BURNUPS ARE TO BE USED THIS CYCLE.	READ0266
С							READ0267
С	19-21	IFEDGE	1	(IS)		=0 INPUT THE EDGE DATA.	READ0268
C				- •		=-1 USE THE EDGE DATA FROM THE	READ0269
С						PREVIOUS CASE FOR THIS CYCLE.	READ0270
С С С С						>0 GENERATE EDGE DATA AUTOMATICALLY	READ0271
С						=1 ASSUME A RING GEOMETRY IN THE	READ0272
С						CORE FOR THE EDGE DATA.	R EA D 0 2 7 3
C						=2 ASSUME A CHECKERBOARD GEOMETRY	READO274
C						FOR THE INTERIOR OF THE CORE.	R EA DO 275
С							R E A D O 276
С	22-24	DUMMY	1				R EA D0277
C							READO278
С	25-30	CYBU	R	(IS)	GWD/NTM	DESIRED (OR GUESSED) CYCLE BURNUP.	READ0279
C C							READ 028 0
С	31-36	BP	R	(IS)		NUMBER OF BP RODS (OR GUESS) IN	READ0281
С						CORE.	READ0282
C							READ0283
C	37-42	EOLK	R	(IS)		DESIRED END OF LIFE K EFF FOR CYCLE.	
C						CAN BE VARIED TO CHANGE FOR EOL	READ0285
C						BORON CONCENTRATION, COASTDOWN OR	
C						EARLY SHUTDOWN.	READO287
С							READ0288

~

C C	43-48	CAPFAC	R	(IS) PI	ERCENT	CAPACITY FACTOR FOR CYCLE, EXCLUDING REFUELING SHUTDOWN.		READO289 READO290
Ċ								R EA DO291
ē	49-54	CYTIM	R	(IS)	DAYS	CYCLE LENGTH, EXCLUDING SHUTDOWN.		READO 292
С						ONLY 2 OF 3 (CYBU, CAPFAC, CYTIM)		R EA DO293
С						NEED BE ENTERED. 1F 3 ENTERED,		READO294
C						CYTIM IS RECALCULATED.		READ0295
С								READ0296
Ċ	55-60	SDLEN	R	(IS)	DAYS	REFUELING SHUTDOWN LENGTH.		READ0297
С				•		IF NOT ENTERED, SET TO 42.		READ0298
c								READ0299
С								READ0300
C	****	FEED DA	TA C	CARDS ()	213,A3	,I3,2F6.0)		R E A D O 30 1
С				INPUT	FOR EA	CH REGION IN CYCLE 1 OR JUST THE FEED		R EA D0302
С						OTHER CYCLES. START WITH THE LOWEST		READ0303
С				ENRICH	MENT R	EGION AND PUT IN ORDER OF INCREASING		READ0304
С				ENRICH	MENT_			READ0305
C								READ0306
С	1-3	NAFEED	I	(IS,I)		NUMBER OF FEED I (OR REGION I IN		R EA D0307
С		NAC1	1	(I)	C	YCLE 1) ASSEMBLIES.		READ0308
С								READ0309
С	4- 6	FTYPE	I	(IS,I)		FUEL TYPE FOR FEED I (OR REGION I		READ0310
С		C 1 TY PE	ĩ	(I)		IN CYCLE 1) CORRESPONDING TO INPUT		READ0311
С						TABLE OF K INF. INTERPOLATION WILL		READ0312
C C C						ONLY BE DONE WITH IDENTICAL FUEL		READ0313
С					Т	YPES.		READ0314
С								READ0315
С	7- 9	FID	A	(IS,I)		ID FOR FEED REGION 2. (IE. 5A,6PU).		READ0316
С		CIID	A	(I)		THIS IS USED FOR LABELING OF OUTPUT.		RZAD0317
С								READO318
С	10-12	DUMMY	1					READO319
С							,	R EA D0320
С	13-18	FRICH	R	(IS,I)	W/0	ENRICHMENT OF FEED I IN CYCLE IS OR	-	R EADO 321
С		C1RICH	ĸ	(1)	W/O	ENRICHMENT OF REGION I IN CYCLE 1.		READ0322
С								READ0323
C	19-24	FLOAD	R	(1S,I)	MTM	LOADING PER ASSEMBLY FOR FEED I IN		READ0324

	CILOAD	R	(I)	MTM	CYCLE IS OR REGION I IN CYCLE 1. IF NOT ENTERED, SET TO ULOAD/NASS.	READO325 READO326
						R EA D0327
						R #AD0 328
* * * *	REGION	SIZE	CARD	(12I6)	(ONLY IF ISHUF(IS)=0)	READ0329
						READ0330
	NAREG	I	(I,IS)		NUMBER OF ASSEMBLIES IN EACH REGION	READ0331
					STARTING WITH THE HIGHEST BURNUP	READ0332
					(INNERMOST) REGION. NREG(IS) ENTRIES,	READ0333
					12 PER CARD.	READ0334
						READO335
						R EA DO336
****	SHUFFLI	E CAR	D (12I	6)	(ONLY IF ISHUF(IS)=0)	READO 337
			-	-	• • • •	READ0338
	ISHUF	I	(I,IS)		SHUFFLE INICIES SPECIFYING PREVIOUS	READ0339
					LOCATION OF EACH REGION(I) IN CYCLE	READO340
					IS. VALUE OF FORM 100*K+L WHERE	READ0341
					K IS THE PREVIOUS CYCLE NUMBER AND	READO342
					L IS THE REGION NUMBER IN THE CYCLE.	READO343
					FOR A FEED REGION, K=O AND L IS THE	READO344
					FEED NUMBER (1,2, OR 3) NREG(IS)	R EA DO345
					ENTRIES, 12 PER CARD.	READ0346
						R EA DO 347
						READO348
****	EDGE CA	RDS	(1116)	(ONLY IF IFEDGE(IS)=0)	READO349
						READ0350
1- 6	JR	1			REGION NUMBER CORRESPONDING TO DATA	READO351
					ON THE WHOLE CARD.	READ0352
						READO 353
	NEDGE	1	(1,K)		SPECIFICATIONS OF THE NUMBER OF EDGES	R EA D0354
	NEDGE		(2,5)		ADJOINING REGION JR IN CYCLE IS.	READO 355
19-24	NEDGE	I	(1,K+1		NEDGE(1,.) IS THE REGION NUMBER	R EADO356
	NEDGE	I	(2,K+1)	ADJOINING REGION JR AND NEDGE(2,.)	READO357
					IS THE NUMBER OF EDGES BETWEEN THEM.	READ0358
	ETC.					READ0359
	ETC.				FOR EDGES ON THE EXTERIOR, SET NEDGE(1,.)=0. NOTE THAT K IS JUST	READ0360

C	A COUNTER. ALSO ONLY NON-ZERO	READ0361
C	NUMBER OF EDGES MUST BE ENTERED AND	R EADO 362
С	ONLY ONE SET FOR EACH REGION PAIR.	R EA DO 36 3
С	5 PAIRS MAY BE ENTERED ON EACH CARD.	
C	NOTE: LAST CARD WITH EDGE DATA SHOULD BE	R EA D0365
C	FOLLOWED BY A BLANK CARD.	READO 366
С		R EA D0367
С		READO 368
С	**** BP FRACTION CARD (12F6.2) (ONLY IF BP(IS) NE 0)	READ0369
Ċ	•	READ0370
С	BPFRAC R (I,IS) FRACTION OF BP RODS IN EACH REGION	READ0371
С	I IN CYCLE IS. NREG(IS) ENTRIES,	READ0372
С	12 ENTRIES PER CARD.	READO373
С		R EA D0374
С		R E A D O 375
С	**** BOC BURNUP CARD (12F6.2) (ONLY IF IBOC=1)	READ0376
С		R E A D O 377
С	BOCBU R (I,IS)GWD/MTM BOC REGION BURNUPS FOR CYCLE IS.	READ0378
С	NREG(IS) ENTRIES, 12 ENTRIES / CARD.	
С		READ0380
С		READ0381
C	**** END OF CYCLE (IS) DATA - REPEAT SET FOR NEXT CYCLE.	R EA DO 382
С	**** AFTER LAST CYCLE DATA SET, PUT A BLANK CARD.	READ0383
C	**** CASES MAY BE STACKED DIRECTLY.	READO384
C		R EA D0385
C		READO386
С		READ0387
	COMMON /TABL1/ NBU, NRICH, BURNUP (20), ENRICH (20), ITYPE (20), BIAS (20),	READ0388
	1 BPWRTH (20), CYBIAS	READ0389
	COMMON /IO/ IO5, IO6, IO7	READ0390
	COMMON /TABL2/ KINF (20,20), ISOTOP (20,20,6), POWFIS (20)	READ0391
	REAL KINF, ISOTOP	R EA D0392
	COMMON /BASIC/ TITLE(20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT, BUDIE-ISO, W. ALPHA, FLIM(2), THAL, POWER, DEBUG	
		READ0394
	LOGICAL DEBUG	READO 395
	COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21),	READ0396

```
NFEED (21), NREG (21), CYLOAD (21), CAPFAC (21), CYTIM (21),
                                                                                         READ0397
     1
      2
                      SDLEN(21), IFEDGE(21)
                                                                                         READ0398
      COMMON /RGDAT/ NEDGE (2, 1000), IEDGE (21), NAREG (15, 21), JSHUF (15, 21),
                                                                                         READ0399
                       BPFRAC(15,21), BOCBU(15,21), EOCBU(15,21),
     1
                                                                                         READ0400
     2
                       IDXREG (15,21), NADIS (15,21)
                                                                                         READ0401
      COMMON /CYCL1/ C1RICH(10), C1TYPE(10), C1ID(10), C1LOAD(10), NAC1(10)
                                                                                         READ0402
      COMMON /FEED / NAFEED (21, 3), FTYPE (21, 3), FLOAD (21, 3), FRICH (21, 3),
                                                                                         READ0403
     1
                      FID(21,3)
                                                                                         READ0404
      INTEGER FTYPE, C1TYPE
                                                                                         READ0405
      COMMON /COST / IFCOST
                                                                                         READ0406
      COMMON /COEF / COFIT (11), TEMP (2), BWRTH, DPWRTH, XEWRTH
                                                                                         READ0407
      COMMON /LOOP/ NA, NW, DELA, DELW
                                                                                         READ0408
      DIMENSION IDUM (24), DUM (12)
                                                                                         READ0409
      DIMENSION B(10)
                                                                                         READ0410
      DATA BLANK / 3H
                                                                                         READ0411
                           1
C
                                                                                         READ0412
      READ (105,901, END=200) TITLE
                                                                                         READ0413
      READ (105,901
CDC
                             ) TITLE
                                                                                         READ0414
      IF (EOF (IO5)) 200,5
CDC
                                                                                         READ0415
CDC 5 CONTINUE
                                                                                         READ0416
С
                                                                                         READ0417
      READ (105,910) NASS, NADIAM, NB, NR, IS, IFCOST, INFIT, IHAL, DEBUG, NW, NA
                                                                                         READ0418
С
                                                                                         READ0419
      READ (105,904) W, ALPHA, ELIM(1), ELIM(2), BUDIF, CYBIAS, HEIGHT,
                                                                                         READ0420
     1
            ULOAD, POWER, DELW, DELA
                                                                                         READ0421
      CALL BUGOUT (8HW , W, 1, 1, 1, 1, 1, 1, 2)
                                                                                         READ0422
      CALL BUGOUT (8HALPHA
                              "ALPHA, 1, 1, 1, 1, 1, 1, 2)
                                                                                         READ0423
      ANASS=NASS
                                                                                         READ0424
      1SO=IS
                                                                                         READ0425
      IF (W.EQ.0.0) W=0.115
                                                                                         READ0426
      IF (ALPHA.EQ.0.0) ALPHA=0.30
                                                                                         READ0427
      IF (ULOAD.EO.0.0) ULOAD=1.0
                                                                                         READ0428
      IF (ELIM(2) \cdot EQ \cdot 0 \cdot 0) = ELIM(2) = 4.5
                                                                                         READ0429
      IF (POWER.EQ.0.0) POWER=1.0
                                                                                         READ0430
      IF (CYBIAS.EQ.0.0) CYBIAS=1.0
                                                                                         READ0431
      IF (NA.EQ.0) NA=1
                                                                                         R EAD0432
```

С	IF (NW-EQ-O) NW=1	READ0433
C		READO434
	IF (NB.LE.O) GO TO 10 READ (105,904) (BURNUP(I),I=1,NB)	READ0435
	NBU = NB	READO436
		READ0437
	IF (HE1GHT.EQ.O.O) READ (IO5,905) (BIAS(I),I=1,NB) READ (IO5,905) (BPWRTH(I),I=1,NB)	READO438
	10 CONTINUE	READ0439
С	IO CONTINUE	READO440
L		READO441
	IF (NB.GE.0) GO TO 13 PEND (TO5 0.05) (P(T) 1-1.5)	READ0442
	READ (105,905) (B(I),I=1,5) DO 12 I=1,NBU	READ0443
	BU = BURNUP(I)	READ0444
	BPWRTH(I)=B(1)*(1.0+B(2)*BU+B(3)*BU**2+B(4)*BU**3+B(5)*BU**4)	READ0445
	BPWRTH(I) = AMAX1(0.0, BPWRTH(I))	READ0446
	12 CONTINUE	R E ADO 4 4 7 R E A DO 4 4 8
	13 CONTINUE	READ0448 READ0449
С		READ0449 READ0450
-	1F (NR.EQ.0) GO TO 35	READ0450
	IF (NR.LT.0) GO TO 15	READ0452
	NR1=1	READ0453
	NRICH=NR	READ0454
	GO TO 20	R EA D0455
	15 CONTINUE	R EADO 456
	NR1=NRICH+1	READ0457
	NRICH=NRICH-NR	READ0458
	20 CONTINUE	READ0459
	READ $(IO5,904)$ $(ENRICH(I), I=NR1, NRICH)$	READ0460
	READ (105,906) (ITYPE(I), I=NR1, NRICH)	READ0461
	READ $(105,904)$ (POWFIS(I), I=NR1, NRICH)	R EA D 0 4 6 2
	DO 25 I=NR1, NRICH	READ0463
С		R EA D0464
	IF (NB.LT.0) GO TO 21	READO465
	READ $(105,905)$ $(KINF(J,I), J=1, NBU)$	READ0466
	GO TO 26	READ0467
С		READ0468

	21	CONTINUE	R EA D 0 4 6 9
		READ $(105,905)$ $(B(J), J=1,8)$	READ0470
		DO 22 $J=1$, NBU	R EA D0471
		BU=BURNUP(J)	READ0472
		A=1.0-B (5) *BU+B (6) *BU **2+B (7) *BU**3+B (8) *BU**4	READ0473
		KINF(J,I) = B(1) * A * (1.0 - B(2) - B(3) * (1.0 + B(4) * BU))	READO474
	22	CONTINUE	READ0475
		CONTINUE	R EA D0476
		CONTINUE	READ0477
С			READ0478
•		IF (ISO.NE.1) GO TO 35	READ0479
		DO 30 I=NR1, NRICH	READ0480
		DU 30 L=1,6	READ0481
		READ $(105,905)$ $(ISOTOP(J,I,L),J=1,NBU)$	READ0482
	30	CONTINUE	READ0483
		CONTINUE	READ0484
С			READ0485
		DO 41 1=1,21	READ0486
	41	ISURCH $(I) = 0$	READ0487
С			READO488
		IF (INFIT.EQ.0) GO TO 40	R EA D0489
		READ (105,904) XEWRTH, DPWRTH, BWRTH, TEMP	R EADO 490
		READ (105,907) COFIT	READ0491
	40	CONTINUE	READ0492
С			READO493
С		CYCLE INPUT ON A CYCLE BY CYCLE BASIS	R EA D 0 4 9 4
С			READ0495
		READ $(IO5, 903, END=150)$ $(IDUM(I), I=1, 8)$, $(DUM(I), I=1, 6)$	R EA D 0 4 9 6
CD	С	READ (105,903) (IDUM (I),I=1,8), (DUM (I),I=1,6)	READO497
CD		IF (EOF(105)) 150,6	R EA D 04 98
		CONTINUE	READO499
		IF (IDUM (1).EQ.0) GO TO 150	READ0500
		IS=IDUM(1)	READ0501
		1 SURCH(1 S) = IDUM(2)	READ0502
		1 SHUF(IS) = I DUM(3)	R EA D0503
		N HEG(IS) = IDUM(4)	READO504

NFEED(IS)=IDUM(5)	R EA D0505
1BOC=IDUM(6)	R E A D O 50 6
IFEDGE (IS) = IDUM(7)	READ0507
CYBU(IS) = DUM(1)	READ0508
BP(IS) = DUM(2)	READ0509
EOLK(IS) = DUM(3)	READ0510
CAPFAC(1S) = DUM(4)	R EAD0511
CYTIM(IS) = DUM(5)	READ0512
SDLEN(IS) = DUM(6)	READ0513
IF $(SDLEN(IS) \cdot EQ \cdot 0 \cdot 0)$ $SDLEN(IS) = 42$.	R EA D 05 1 4
C	READ0515
IF (IS.GT.1) GO TO 50	READ0516
NR = NREG(IS)	READ0517
DO 45 I=1, NR	READ0518
READ (105,909) NAC1(I), C1TYPE(I), C1ID(I), IDUM(1), C1RICH(I),	R EA D0519
1 C1LOAD (I)	READ0520
IF $(C1LOAD(I) \cdot EQ \cdot 0 \cdot 0)$ C1LOAD $(I) = ULOAD/ANASS$	R EA D0521
C	READ0522
CDC IF (C1ID(I).EQ.BLANK) ENCODE (3,903,C1ID(I)) I	R EA D0523
NF = NR	READ0524
45 CONTINUE	READ0525
GO TO 60	READ0526
С	READ0527
50 CONTINUE	R EA D 0 5 2 8
NR=NFEED(IS)	R EAD0529
IF (NR.EQ.U) GO TO 60	READ0530
DO 55 I=1, NR	READ0531
READ (105,909) NAFEED(IS,1), FTYPE(IS,1), FID(IS,1), IDUM(1),	R EA D0532
1 FRICH (IS, I), FLOAD (IS, I)	READ0533
NF = NF + 1	READ0534
C	READ0535
CDC IF (FID (IS,I).EQ.BLANK) ENCODE (3,903,FID(IS,I)) NF	READ0536
IF (FLOAD (IS, I) $-EQ.O.O$) FLOAD (IS, I) = ULOAD/ANASS	READ0537
55 CONTINUE	READ0538
60 CONTINUE .	READ0539
C	READ0540

	NB=NREG (IS)	READ0541
	IF (ISHUF(IS).NE.0) GO TO 65	READ0542
	1F (IS.EQ.1) GO TO 65	READ0543
	READ (IO5,906) (NAREG (I, IS), I=1, NR)	READ0544
	READ (105,906) (JSHUF (1,1S), I=1, NR)	READ0545
65	CONTINUE	READ0546
		READO547
	1 N=0	R EA D0548
	IF (IS.GT.1) $IN=IEDGE(IS-1)$	READ0549
	IN2=1EDGE(IS)	READ0550
	IF (IFEDGE(IS).EQ1) GO TO 105	READ0551
	IF $(1N2.EQ.IN)$ GO TO 70	READ0552
	I N 4 = I N + 1	READ0553
	DO 68 J=IN4, IN2	READ0554
	NEDGE $(1, J) = 0$	REA D0555
68	NEDGE(2, J) = 0	READ0556
	CONTINUE	R EA D 0 5 5 7
	IF (IFEDGE(IS).NE.0) GO TO 105	READ0558
	IN3=IEDGE(21)	READ0559
	READ $(105,906)$ JR, $(1DUM(I), I=1, 10)$	READ0560
	1F (JR.EQ.0) GO TO 105	READ0561
	N = 0	READ0562
	DO 75 $J=1,9,2$	READ0563
	1F (IDUM(J).GT.O.OR.IDUM(J+1).GT.O) N=N+1.	R EA D0564
75	CONTINUE	READ0565
	IF (IN+N.LE.IN2) GO TO 95	READ0566
	N=N-(IN2-IN)	READ0567
	IF (IN3.EQ.IN2) GO TO 85	READ0568
	I N4 = I N2 + 1	READ0569
	DO 80 $J=1N4$, $IN3$	READ0570
	K=1N3-J+IN4	READ0571
	NEDGE $(1, K+N) = NEDGE (1, K)$	READ0572
	NEDGE $(2, K+N) = NEDGE (2, K)$	R EA D0573
68	CONTINUE	READ0574
	CONTINUE	READ0575
<u> </u>	IN3 = IN3 + N	READ0576

•

.

С

		1 N 2 = 1 N 2 + N	READ0577
		DO 90 J=15,21	READ0578
	90	IEDGE(J) = IZDGE(J) + N	READ0579
		CONTINUE	READO580
	15	DO 100 $J=1,9,2$	READ0581
		1F (IDUM (J).EQ.O.AND.IDUM (J+1).EQ.O) GO TO 100	READ0582
		IN = IN + 1	READ0583
		NEDGE $(1, IN) = 100 \times IDUM(J) + JR$	READ0584
		NEDGE(2,IN) = IDUM(J+1)	READ0585
	100	CONTINUE	READ0586
		GO TO 70	R EA D0587
	105	CONTINUE	READ0588
		CALL BUGOUT (8HIEDGE , IEDGE, 1, 21, 21, 1, 1, 1, 1)	READ0589
		CALL BUGOUT (8HNEDGE , NEDGE, 1, 2, 2, 1, IEDGE (21), 1000, 1)	READ0590
С			READ0591
		IF (BP(IS).EQ.0.0) GO TO 110	READ0592
		READ (105,904) (BPFRAC(I, IS), I=1, NR)	READ0593
	110	CONTINUE	READ0594
С			READ0595
С		BOC REG BURNUP	R EA D 05 96
С			R E A D O 597
		IF (IBOC.EQ1) GO TO 120	READ0598
		DU 115 I=1,15	READ0599
	115	BOCBU $(I, IS) = 0.0$	READ0600
		1F (IBOC.NE.1) GO TO 120	READ0601
		READ (105,904) (BOCBU(I,1S), I=1, NR)	READ0602
	120	CONTINUE	READ0603
С		Ň	READ0604
C		END OF CYCLE DATA	READ0605
С			READ0606
		GO TO 40	READ0607
	200	CONTINUE	READ0608
		WRITE (106,902)	READ0609
		CALL EXIT	READ0610
	150	CONTINUE	READO611
		RETURN	READ0612

С		READO613
901	FORMAT (20A4)	READO614
902	FORMAT (1H1,9X,26HLAST CASE COMPLETED - EXIT)	READO615
903	FORMAT (813,8E0.0)	READ06 16
904	FORMAT (12F6.2)	READ0617
905	FORMAT (6E12.5)	R EADO6 18
906	FOR MAT (1216)	R EA DO6 1 9
907	' FORMAT (6E12.5)	R E ADO 6 20
909	FORMAT (213,A3,13,2F6.0)	R EA D 06 2 1
910) FORMAT (2413)	READO622
С		READO623
	END	READ0624

. .

SUBROUTINE PREP	PREPOOD
	PREP0002
SUBROUTINE PREP GENERATES DEFAULT OR AUROMATED DATA AND SETUP	DEED0003
FOR PROBLEM EXECUTION .	PREPOOOL
	PREPOODS
COMMON /TABL1/ NBU, NRICH, BURNUP (20), ENRICH (20), ITYPE (20), BIAS (20),	PREPOODE
BPWRTH (20), CYBIAS	PREP0007
COMMON /10/ 105, 106, 107	PREPOOOS
COMMON /TABL2/ KINF (20,20), ISOTOP (20,20,6), POWFIS (20)	PREPOUOS
REAL KINF, ISOTOP	PREPOO1(
COMMON /BASIC/ TITLE (20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT,	PREPO01
BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG	PREP0012
LOGICAL DEBUG	PREPOO13
COMMON /CYDAT/ ISURCH(21), ISHUP(21), BP(21), CYBU(21), EOLK(21),	PREPO014
NFEED(21), NREG(21), CYLOAD(21), CAPFAC(21), CYTIM(21),	PREPOO15
SDLEN(21), I FEDGE(21)	PREP0016
COMMON /RGDAT/ NEDGE(2,1000), IEDGE(21), NAREG(15,21), JSHUF(15,21),	PREP0017
BPFRAC (15,21), BOCBU (15,21), EOCBU (15,21),	PREP0018
IDXREG (15,21), NADIS (15,21)	PREP0019
COMMON /CYCL1/ C1RICH(10),C1TYPE(10),C1ID(10),C1LOAD(10),NAC1(10)	PREP0020
COMMON /FEED / NAFEED (21, 3), FTYPE (21, 3), FLO AD (21, 3), FRICH (21, 3),	PREP0021
FID (21,3)	PREPOO22
INTEGER FTYPE, C1TYPE	PREP0023
COMMON /COST / IFCOST	PREP0024
COMMON /COEF / COFIT(11), TEMP(2), BWRTH, DPWRTH, XEWRTH	PREP0025
COMMON /EDGE/ IDGE(15,20)	PREP0026
	PREP0027
DIMENSION DBIAS(20)	PREP0028
DIMENSION NCDATA (4, 10)	PREP0029
DATA NCDATA / 121, 12,32,0, 157,15,36,0, 193,15,44,0, 217,17,44,0,	PREPOO3(
24*0 /	PREP0031
DATA DHGHT, DBIAS / 12.0, 1.00567, 1.007425, 1.00902, 1.0099,	PREP0032
16*1.01023 /	PREP003
	PREP0034
IERR=0	PREP0035
GENERATE AXIAL LEAKAGE BIAS FROM DEFAULT	PREP0036

C C C C

С

С

С

		IF (HEIGHT.EQ.0.0) GO TO 10	PREP0037
		PACTOR = (DHGHT+0.5) / (HEIGHT+0.5)	PREPOO38
		DO 5 I=1, NBU	PREP0039
	5	BIAS(I) = FACTOR*(DBIAS(I) - 1.0) + 1.0	PR EP 0040
	10	CONTINUE	PREP0041
С		SET UP CYCLE 1	PREP0042
		NR=NREG (1)	PREPOO43
		DO 15 I=1, NR	PREP0044
		NAREG(1, 1) = NAC1(1)	PREP0045
		JSHUF(I,1)=I	PREPOO46
		IDXREG(I, 1) = 100 + I	PREPOO 47
	15	CONTINUE	PREP0048
		IC1=1	PREP0049
		NC1=NAREG(I,1)	PREP0050
		IF $(NC1.EQ.1)$ IC1=2	PREP0051
		IF $(NC1-EQ.1)$ $NC1=NC1+NAREG(I,2)$	PREP0052
С		GENERATE SHUFFLE ARRAY	PREPO053
		JC1=0	PREP0054
		DO 60 $I=2,20$	PREP0055
		IF (ISURCH(I).EQ.0) GO TO 60	PREP0056
		IF (ISHUF(I).LE.O) GO TO 49	PREPOO57
		NCA = 0	PREP0058
		1CA = MOD (1SHUP (1) + 1, 2)	PREP0059
		IF (JC1.GE.NC1) ICA=0	PREP0060
		NF = NF EED(I)	PREP0061
		1F (NF.EQ.0) GO TO 21	PREP0062
		DO 20 J=1, NF	PREPOO63
		NCA=NCA+NAFEED (I,J)	PREP0064
	21	CONTINUE	PREP0065
		N R = N F	PREPOO66
		NR 1=0	PREP0067
		IF (NCA+1CA.GE.NASS) GO TO 30	PREP0068
		NRR = NREG(I-1)	PREP0069
		DO 25 J=1, NRR	PREP0070
		JJ = NRR - J + 1	PREP0071
		NCA = NCA + NAREG (JJ, I-1)	PREP0072

	N R 1=NR 1+1	PREP0073
	NR = NR + 1	PREP0074
	IF (NCA+ICA.GE.NASS) GO TO 30	PREP0075
25	CONTINUE	PREP0076
30	CONTINUE	PREPO077
	NREG $(I) = NR + ICA$	PREP0078
	NCA = 0	PREP0079
	NR=NR+ICA	PREP0080
	IF (NF.EQ.0) GO TO 36	PREPOO81
	$DO_{35} J = 1, NF$	PREP0082
	JJ = NR - J + 1	PREP0083
	NAREG $(JJ, I) = NAFEED (I, NF-J+1)$	PREPOO84
	NCA = NCA + NAFEED (I, NF-J+1)	PREP0085
	JSHUF(JJ,I) = NF - J + 1	PREP0086
	IDXREG(JJ,I) = 100 *I + NF - J + 1	PREP0087
	CONTINUE	PREP0088
36	CONTINUE	PREP0089
	IF (NR1.EQ.0) GO TO 41	PREP0090
	DO 40 $J = 1, NR 1$	PREP0091
	J 1 = NR - NF - J + 1	PREP0092
	J 2 = NRR - J + 1	PREP0093
	JSHUF(J1,I) = 100 * (I-1) + J2	PREP0094
	IDXREG(J1,I) = IDXREG(J2,I-1)	PREP0095
	NAREG $(J1, I) = MINO (NAREG (J2, I-1), NASS-NCA-ICA)$	PREP0096
	NCA = NCA + NAREG (J1, I)	PREP0097
	CONTINUE	PREP0098
41	CONTINUE	PREPÓO99
	IF (ICA.EQ.0) GO TO 45	PREP0100
	NAREG $(1, I) = 1$	PREP0101
	JC1=JC1+1	PREP0102
	J=1	PREP0103
	IF $(JC1.GT.NAREG(1,1))$ J=2	PREP0104
	J SHUF (1, I) = 100 + J	PREP0105
	IDXREG(1,I) = 100+J	PREP0106
45	CONTINUE	PREP0107
	GO TO 56	PREP0108

•

			PREP0109
	49	CONTINUE NR=NREG (I)	PREP0110
		DU 55 J=1, NR	PREP0111
		JK = JSHUF(J,I)	PREP0112
			PREP0113
		JJ = JK / 100	PREP0114
		K=MOD(JK,100)	PREP0115
		IF (JJ - EQ. 0) GO TO 50	PREPO 116
		IDXREG(J,I) = IDXREG(JJ,K)	PREPO117
	EA	GO TO 55	PREPOIL
		IDXREG(J,I) = 100 * I + K	PREPO119
		CONTINUE	PREPO 120
	20	CONTINUE	PREPO121
		CALL BUGOUT (8HJSHUF , JSHUF, 1, NR, 15, I, I, 21, 1) CALL BUGOUT (8HIDXREG , IDXREG, 1, NR, 15, I, I, 21, 1)	PREPO122
	<i>c</i> 11	CALL BUGOUT (8HIDXREG , IDXREG, 1, NR, 15, I, I, 21, 1) CONTINUE	PREP0123
С	bυ	SHUFFLE FOR EQUILIBRIUM CYCLE	PREPO124
Ļ		IF (ISHUF(21).LE.O.OR.ISURCH(21).EQ.O) GO TO 90	PREP0125
		NP = NP EED (21)	PREP0126
		N = 0	PREP0127
		NR=0	PREP0128
	62	CONTINUE	PREP0129
	02	DO 65 I=1, NF	PREP0130
		NR = NR + 1	PREP0131
		NA=NA+NAFEED(21,I)	PREP0132
		IF (NA.GE.NASS) GO TO 70	PREP0133
	65	CONTINUE	PREP0134
	05	IF (NR.LT.15) GO TO 62	PREP0135
		CALL EXIT	PREP0136
	70	CONTINUE .	PREP0137
	10	NREG(21) = NR	PREP0138
		NA=0	PREP0139
	70		PREP0140
	12	DO 75 $l=1$, NF	PREP0141
		NAREG $(NR, 21) = MINO (NAFEED (21, I), NASS-NA)$	PREP0142
		NA=NA+NAREG(NR, 21)	PREP0143
		IDXREG(NR, 21) = 2100 + I	PREP0144

		IF $(NR+1-1.EQ.NREG(21))$ JSHUP $(NR_21) = I$	PREP0145
		IF $(NR+I-1.LT.NREG(21))$ JSHUP $(NR, 21) = 2100+NR+NP$	PREP0146
		NR = NR - 1	PREPO147
		IF (NA.GE.NASS) GO TO 80	PREP0148
	75	CONTINUE	PREP0149
		GO TO 72	PREP0150
	80	CONTINUE	PRBP0151
		N &= N R E G (21)	PREP0152
		CALL BUGOUT (8HJSHUF , JSHUF, 1, NR, 15, 21, 21, 21, 1)	PREP0153
		CALL BUGOUT (8HIDXREG , IDXREG, 1, NR, 15, 21, 21, 21, 1)	PREP0154
	90	CONTINUE	PREP0155
		IF (ISHUF(21).NE.O.OR.ISURCH(21).EQ.O) GO TO 100	PREP0156
		NF=NFEED(21)	PREP0157
		NR = NREG (21)	PREP0158
		DO 98 JR=1,NR	PREP0159
		JK = JSHUF(JR, 21)	PREP0160
	92	CONTINUE	PREP0161
		IF (JK.LT.100) GO TO 93	PREP0162
		K=MOD (JK, 100)	PREP0163
		IF (K.GT.NR.OR.K.LT.1) GO TO 98	PREP0164
		JK=JSHUF(K,21)	PREP0165
	6.2	GO TO 92	PREP0166
	93	CONTINUE	PREP0167
	0.0	IDXREG (JR, 21) = 2100 + JK	PREP0168
	98	CONTINUE	PREPO169
	100	CALL BUGOUT (8HIDXREG, IDXREG, 1, NR, 15, 21, 21, 21, 1)	PREP0170
C	100	CONTINUE	PREP0171
C			PREP0172
C		GENERATE EDGE DATA	PREP0173
		JJJJ=0	PREP0174
		DO 105 $I=1,10$	PREP0175
	105	IF (NASS.EQ.NCDATA(1,I).AND.NADIAM.EQ.NCDATA(2,I)) JJJJ=I	PREP0176
	10.5	CONTINUE	PREP0177
		DO 500 IC=1,21 IR (1808CH(IC) RO 0) CO TO 500	PREP0178
		IF (1SURCH(IC).EQ.0) GO TO 500 IF (IFEDGE(IC).LE.0) GO TO 500	PREP0179
		TL (TLEDGE (TC) = PE= A) GO TO DAA	PREP0180

		IF (JJJJ.GT.0) GO TO 110	PREP0181
		WRITE (106,901) NASS, NADIAM	PREP0182
		CALL EXIT	PREP0183
	110	CONTINUE	PREP0184
		DO 95 $I=1, 15$	PREP0185
		DO 95 J=1, 20	PREP0186
	95	IDGE(I,J)=0	PRED0182
C		NUMBER OF EXTERIOR EDGES = N1	PREP0188
		N1=4*NADIAM	PREP0189
С		NUMBER OF ASSEMLLIES ON PERIPHERY = N2	PREP0190
		N2=NCDATA (3,JJJJ)	PREP0191
С		NUMBER OF ASSEMBLIES AT INSIDE CORNER OF PERIGHERY = N3	PREP0192
		$N_{3} = N_{1} - N_{2} - 4$	PREP0193
С		NUMBER OF PERIPHERAL ASSEMBLIES WITH 2 EDGES ON EXTERIOR = N4	PREP0194
		N 4 = N 1 - N 2	PREP0195
С			PREP0196
Ĉ		IDGE(,17) - NO. ASSEBLIES WITH 2 EDGES ON PERIPHERY	PREP0197
C		IDGE (,18) - NO. ASSEBLIES WITH 1 EDGES ON PERIPHERY	PR EP0198
С		IDGE(,19) - NO. ASSEMBLIES ON INSIDE CORNER OF PERIPHERY	PREP0 199
С		IDGE (,20) - NO. OF EDGES USED SO FAR	PREPO200
Ċ			PREP0201
C		DETERMINE PERIPHERAL EDGES	PREP0202
		NR=NREG (IC)	PREP0203
		IR = NR	PREP0204
		NA=NAREG (NR, IC)	PREP0205
		NN = N4	PREP0206
С		2 EDGES ON PERIPHERY	PREP0207
	115	CONTINUE .	PREP0208
		NUSE=MINO (NA, NN)	PREP0209
		IDGE(IR, 20) = IDGE(IR, 20) + 2 * NUSE	PR EP 0210
		IDGE (IR, 16) = IDGE (IR, 16) + 2*NUSE	PREP0211
		IDGE(IR, 17) = NUSE	PREP0212
		NN = NN - NUSE	PREP0213
		NA = NA - NUSE	PREPO214
		I 2P = IR	PREP0215
		IF (NA.GT.U) GO TO 120	PREP0216

		IR=IR-1	PREP0217
		NA=NAREG (IR,IC)	PREP0218
		IF (NN.GT.0) GO TO 115	PREP0219
	120	CONTINUE	PREP0220
		CALL BUGOUT (8HIDGE-A , IDGE, 1, NR, 15, 16, 20, 20, 1)	PREP0221
С		I EDGE ON PERIPHERY	PREP0222
		NN = N2 - N4	PREP0223
		J1P=IR	PREP0224
	125	CONTINUE	PREP0225
		NUSE=MINO(NA, NN)	PREP0226
		IDGE(IR, 20) = IDGE(IR, 20) + NUSE	PREP0227
		IDGE(IR, 16) = IDGE(IR, 16) + NUSE	PREP0228
		IDGE (IR, 18) = NUSE	PREP0229
		I1P=IR	PREP0230
		N N = N N - N USE	PREP0231
		NA = NA - NUSE	PREP0232
		IF (NA.GT.O) GO TO 130 ·	PREP0233
		IR = IR - 1	PREP0234
		NA=NAREG (IR,IC)	PREP0235
		IF (NN.GT.0) GO TO125	PREP0236
	130	CONTINUE	PREP0237
		CALL BUGOUT (8HIDGE-B , IDGE, 1, NR, 15, 16, 20, 20, 1)	PREP0238
		NA = IDGE(I1P, 18)	PREP0239
		1R=11P	PREP0240
		DO 140 I=12P, NR	PREP0241
		J=NR-1+I2P	PREP0242
		NN = IDGE(J, 17)	PREP0243
	135	NUSE=MINO(2*NN,NA)	PREP0244
		IDGE (IR, 20) = IDGE (IR, 20) + NUSE	PREP0245
		IDGE(IR,J) = IDGE(IR,J) + NUSE	PREP0246
		1DGE (J , 20) = IDGE (J , 20) + NUSE	PREP0247
		IDGE(J,IR) = IDGE(J,IR) + NUSE	PREP0248
		NN = NN - NUSE/2	PREP0249
		NA = NA - NUSE	PREP0250
		IF (NA.GT.O) GO TO 140	PREP0251
		IR=IR-1	PREP0252

	NA = IDGE(IR, 18)	PREP0253
	IF (NA.EQ.0) GO TO 140	PREP0254
	IF (NN.GT.0) GO TO 135	PREP0255
140	CONTINUE	PREP0256
	CALL BUGOUT (8HIDGE-C , IDGE, 1, NR, 15, 1, NR, 20, 1)	PREP0257
	CALL BUGOUT (8HIDGE-C , IDGE, 1, NR, 15, 16, 20, 20, 1)	PHEP0258
	IF (J1P.GT.I1P) GO TO 150	PREP0259
	NIN=IDGE(I1P, 18)	P R E P 0 2 6 0
	NUSE=3*NIN-IDGE(I1P,20)+4*IDGE(I1P,17)-2*N3	PREP0261
	IF (NUSE-LE-0) GO TO 180	PREP0262
	IDGE (I1P, I1P) = IDGE (I1P, I1P) + NUSE	PREP0263
	IDGE (I1P,20) = IDGE (I1P,20) + NUSE	PREP0264
	GO TO 180	PREP0265
150	CONTINUE	PREP0266
	1R = J1P	PREP0267
	JH = IR - 1	PREP0268
152	CONTINUE	PREP0269
	NN = IDGE (IR, 20)	PREP0270
	NIN=IDGE(IR, 18)	PREP0271
	NEED= 4*NIN-NIN-NN+4*IDGE (IR, 17)	PREP0272
157	CONTINUE	PREP0273
	J N= 0	PREP0274
	DO 160 I=1,16	PREP0275
160	JN=JN+IDGE(JR,I)	PREP0276
	JNEED=3*IDGE(JR, 18)-JN+4*IDGE(JR, 17)	PREP0277
161	NUSE=MINO (NEED, JNEED)	PREP0278
	IDGE(IR, JR) = IDGE(IR, JR) + NUSE	PREP0279
	IDGE(IR,20) = IDGE(IR,20) + NUSE	PREP0280
	IDGE(JR, 20) = IDGE(JR, 20) + NUSE	PREP0281
	IDGE (JR, IR) = IDGE (JR, IR) + NUSE	PREP0282
	JNEED=JNEED-NUSE	PREP0283
	NEED=NEED-NUSE	PREP0284
	IF (IR.EQ.JR) GO TO 180	PREP0285
	IF (JNEED.GT.O) GO TO 165	PREP0286
	NEED=JNEED	PREP0287
	IK=JR	PREP0288

		JR=MAXO(JR-1,I1P)	PREP0289
		GO TO 157	PREP0290
	165	CONTINUE	PREP0291
		IF (NEED.EQ.0) GO TO 170	PREP0292
		IF (JR.EQ.I1P) GO TO 166	PREP0293
		JR=JR-1	PREP0294
		GO TO 157	PREP0295
	166	CONTINUE	PREP0296
		JR=IR	PR EP 0 297
		JNEED=NEED	PREP0298
		GO TO 161	PREP0299
	170	CONTINUE	PREP0300
		IF (JR.EQ.I1P) GO TO 180	PREP0301
		IR=JR-1	PREP0302
		JR=MAXO (IR-1,I1P)	PREP0303
		GO TO 152	PREPO304
		CONTINUE	PREP0305
	185	CONTINUE	PREP0306
		CALL BUGOUT (8HIDGE-D ,IDGE, 1, NR, 15, 16, 20, 20, 1)	PREP0307
		CALL BUGOUT (8HIDGE-D , IDGE, 1, NR, 15, 1, NR, 20, 1)	PREP0308
С		ASSEMBLY ON INSIDE CORNER OF PERIPHERY	PREP0309
		N N = N 3	PREP0310
		JIC=NR+1	PREP0311
		DO 190 I=1, NR	PREP0312
		J=NR-I+1	PREP0313
		NA = NAREG (J, IC) - IDGE (J, 17) - IDGE (J, 18)	PREP0314
		IF $(NA \cdot EQ \cdot 0)$ JIC=J	PREP0315
		IF (NA.EQ.0) GO TO 190	PREP0316
		NUSE=MINU(NN, NA)	PREP0317
		IDGE (J, 19) = NUSE	PREP0318
		IIC=J	PREP0319
		NN = NN - NUSE	PREP0320
		IF (NN.EQ.0) GO TO 195	PREP0321
		CONTINUE	PREP0322
	195	CONTINUE	PREP0323
		JIC=JIC-1	PREP0324

-

Ģ

DO 210 1=IIC, JIC	PREP0325
J=JIC-I +IIC	PREP0326
NN=IDGE (J, 19)	PREP0327
IF (NN.EQ.0) GO TO 210	PREP0328
JR = NR	PREP0329
200 NA=4*IDGE (JR, 17) + 3*IDGE (JR, 18) + 2*IDGE (JR, 19) - IDGE (JR, 20) *	PREP0330
IF (NA.LE.1) GO TO 205	PREP0331
IF $(JR.EQ.J)$ NA=NA/2	PREP0332
NUSE=MINO(NA/2,NN)	PREP0333
NUSE=NUSE*2	PREP0334
IDGE(JR,J) = IDGE(JR,J) + NUSE	PREP0335
IDGE(JR, 20) = IDGE(JR, 20) + NUSE	PREP0336
IDGE (J, 20) = IDGE (J, 20) + NUSE	PREP0337
IDGE(J,JR) = IDGE(J,JR) + NUSE	PREP0338
NA=NA-NUSE	PREP0339
NN = NN - NUSE/2	PREP0340
IF (NA.GT.1) GO TO 210	PREP0341
205 JR = JR - 1	PREP0342
IF (IDGE(JR,20).EQ.0) GO TO 210	PREPO343
GO TO 200	PREP0344
210 CONTINUE	PR EP 0345
CALL BUGOUT (8HIDGE-E ,IDGE, 1, NR, 15, 1, NR, 20, 1)	PREPO346
CALL BUGOUT (8HIDGE-E , IDGE, 1, NR, 15, 16, 20, 20, 1)	PREP0347
C ASSEMBLIES ON INTERIOR IN RINGS	PREP0348
IF (IFEDGE(IC).NE.1) GO TO 300	PREP0349
IRING=1	PREP0350
NRING=1	PREP0351
KRING=0	PREP0352
IC1 = MOD (NASS, 2)	PREP0353
NEI = 4*(2*(1-IC1)+IC1)	PREP 0354
1F (IC1.EQ.0) NRING=4	PREP0355
NK=NRING	PREP0356
DO 260 JR=1, NR	PREP0357
NA=NAREG (JR,IC) -IDGE (JR, 17) - IDGE (JR, 18) - IDGE (JR, 19)	PREP0358
215 CONTINUE	PREP0359
IF (NA.EQ.0) GU TO 260 ·	PREP0360

	NUSE=MINO(NRING,NA)	PREP0361
	NRING=NRING-NUSE	PREP0362
	NA=NA-NUSE	PREP0363
	IF (KRING.GT.O) GO TO 220	PREP0364
	NE=MINO (NEI,4*NUSE)	PREP0365
	IDGE (JR, JR) = 1DGE (JR, JR) + 4 * NUSE - NE	PREP0366
	$IDGE(JR, 20) = IDGE(JR, 20) + 4 \times NUSE - NE$	PREP0367
	KRING=IRING	PREP0368
	GO TO 250	PREP0369
220	CONTINUE	PREP0370
	IF (KRING.EQ.IRING) GO TO 230	PREP0371
С	PREVIOUS TIME FILLED LAST RING - INCREASE EDGES BY 8	PREP0372
	NEK=4*(2*(KRING-IC1)+IC1)	PREP0373
	NEI=4*(2*(IRING-IC1)+IC1)	PREP0374
	NE1=MINO (NUSE, NEK)	PREP0375
	NE3=MINO(NEI, NUSE+8)	PREP0376
	$NE 2= 4 \times NUS E - NE 1 - NE 3$	PREPO377
	IDGE (JR, JR) = IDGE (JR, JR) + NE2	PREP0378
	IDGE(JR, 20) = IDGE(JR, 20) + NE2 + NE1	PREPO379
	DO 225 J=1, JR	PREP0380
	NE=4*NAREG(J,IC)-IDGE(J,20)	PREPO381
	NE=MINO (NE, NE1)	PREP0382
	IDGE(JR, J) = IDGE(JR, J) + NE	PREP0383
	IDGE(J, JR) = IDGE(J, JR) + NE	PREP0384
	1DGE(J, 20) = IDGE(J, 20) + NE .	PREP0385
	NE1 = NE1 - NE	PREP0386
	NEK=NEK-NE	PREP0387
225	CONTINUE	PREP0388
	GO TO 250	PREP0389
230	CONTINUE	PREP0390
C	FILL IN SAME RING AS BEFORE	PREP0391
	NE1=MINO(NUSE, NEK)	PREP0392
	NE3=NE1	PREP0393
	NE2=4*NUSE-NE1-NE3	PREP0394
	NEK=NEK-NE1	PREP0395
	1 DGE (JR, JR) = IDGE (JR, JR) + NE2	PREP0396

.

•

		IDGE(JR,20) = IDGE(JR,20) + NE2 + NE1	PREP0397
		DO 235 $J=1, JR$	PREP0398
		NE=4*NAREG(J,IC)-IDGE(J,20)	PREP0399
		NE=MINO(NE,NE1)	PREP0400
		IDGE(JR,J)=IDGE(JR,J)+NE	PREP0401
		IDGE(J,JR) = IEGE(J,JR) + NE	PREP0402
		IDGE(J,20) = IDGE(J,20) + NE	PREP0403
		NE1 = NE1 - NE	PREP0404
		NEK=NEK-NE	PREP0405
	235	CONTINUE	PREP0406
	250	CONTINUE	PREP0407
		CALL BUGOUT (8HIDGE-F , IDGE, 1, NR, 15, 1, NR, 20, 1)	PREP0408
		IF (NHING.GT.O) GO TO 260	PREP0409
		KRING=IRING ·	PREP0410
		IKING=IKING+1	PREP0411
		N RING=2*(IRING-1)*(IRING+2)+1	PREP0412
		IF (IC1.EQ.1) GO TO 255	PREP0413
		ARING=3.14*IRING**2+3.0	PREPO414
		NRING=0.125*ARING	PREP0415
		NRING=8*NRING	PREP0416
	255	CONTINUE	PREP0417
		NRING=NRING-NK	PREP0418
		N K= N K + N R I NG	PR EP 04 19
		IF (NA.GT.0) GO TO 215	PREP0420
		CONTINUE	PREP0421
	265	CONTINUE	PREP0422
С		COMBINE INTERIOR TO PERIPHERY	PREP0423
		JR=1	PREP0424
		NA1=4*NAREG(JR,IC)-IDGE(JR,20)	PREP0425
		DO 280 J=1, NR	PREP0426
		JJ = NR - J + 1	PREP0427
		NA2=4*NAREG (JJ,IC)-IDGE (JJ,20)	PREP0428
		IF (NA2.EQ.0) GO TO 280	PREP0429
	270	IF (NA1.GT.0) GO TO 275	PREP0430
		JR = JR + 1	PREP0431
		IF (JR.GT.NR) GO TO 280	PREP0432

		٠	
		NA1=4* NAREG $(JR, IC) - IDGE (JR, 20)$ GO TO 270	PREP0433 PREP0434
	076		
	275		PREP0435
		NUSE=MINO(NA1, NA2)	PREP0436
		NA1=NA1-NUSE	PREP0437
			PREP0438
		IF (JJ.EQ.JR) NUSE=NUSE/2	PREP0439
		IDGE (JJ, JR) = IDGE (JJ, JR) + NUSE	PREP0440
		1 DG E (JJ, 20) = I DG E (JJ, 20) + NUSE	PREP0441
		IDGE(JR, 20) = IDGE(JR, 20) + NUSE	PREP0442
		1 DGE (JR, JJ) = I DGE (JR, JJ) + N USE	PR EP 0 4 4 3
		IF (NA2.EQ.0) GO TO 280	PREP0444
	200	IF (NA1.EQ.0) GO TO 270	PREP0445
	280	CONTINUE	PREP0446
		CALL BUGOUT (8HIDGE-G , IDGE, 1, NR, 15, 1, NR, 20, 1)	PREP0447
		CALL BUGOUT (8HIDGE-G , IDGE, 1, NR, 15, 16, 20, 20, 1)	PREP0448
	200	GO TO 450	PREP0449
<i>.</i> .	300	CONTINUE	PREP0450
С			PREP0451
C		CHECKERBOARD PATTER	PREP0452
C			PREP0453
		NRM = 1	PREP0454
		IF (NAREG(1, IC) . NE. 1) GO TO 305	PREP0455
		NRM=2	PREP0456
		IDGE(3, 1) = 4	PREP0457
		IDGE(1,3) = 4	PREP0458
		I DG E (1, 20) = 4	PREP0459
		IDGE(3, 20) = IDGE(3, 20) + 4	PREP0460
	305	CONTINUE	PREP0461
C		NP= NO. EDGES LEFT ON PERIPHERAL ASSEMBLIES	PREP0462
		N P =0	PREP0463
		NR1=NR	PREP0464
		DO 310 JR=1,NR	PREP0465
		IR=NR-JR+1	PREP0466
		K=IDGE(IR, 17) +IDGE(IR, 18) +IDGE(IR, 19)	PREP0467
		IF (K.EQ.0) GO TO 315	PREP0468

		NE1=IR	PREP0469
		NP=NP+4*K-IDGE(IR,20)	PREP0470
	310	CONTINUE	PREP0471
		CONTINUE	PREP0472
	515	IF $(K \cdot EQ \cdot NAREG(NR1, IC))$ $NR1 = NR1 - 1$	PREP0473
		NR2=MAXO(NRM, NR1-1)	PREP0474
C	•	NR1- NR2 REGIONS IN INTERIOR TO FACE PERIPHERY ASSEMBLIES	PREP0475
		NSUM=0	PREP0476
	520	DO 325 K=NR2,NR1	PREP0477
	325	NSUM=NSUM+NAREG(K,IC)-IDGE(K,17)-IDGE(K,18)-IDGE(K,19)	PkEP0478
	323	IF (NSUM.GE.NP/2) GO TO 330	PREP0479
		IF (NR2.EQ.1) GO TO 330	PREP0480
		NR2 = NR2 - 1	PREP0481
		GO TO 320	PREP0482
	330	CONTINUE	PREP0483
		DO 350 IR=NR1, NR	PREP0484
		K=IDGE(IR, 17) +IDGE(IR, 18) +IDGE(IR, 19)	PREP0485
		IF (K. EQ. 0) GO TO 350	PREP0486
		NE=4 * K - IDGE(IR, 20)	PREP0487
		A=NE	PREP0488
		DO 340 JR=NR2, NR1	PREP0489
		B=NAREG (JK, IC) -IDGE (JR, 17) -IDGE (JR, 18) - IDGE (JR, 19)	PREP0490
		B=B/NSUM	PREP0491
		NUSE=A*B+0.5	PREP0492
		N 1=4*NAREG (JR, IC) - IDGE (JR, 20)	PREP0493
		IF (JR.EQ.NR1) NUSE=NE	PREP0494
		NUSE=MINO (NE, NUSE)	PREP0495
		IF (IR. EQ. JR) $N1 = N1/2$	PREP0496
		NUSE=MINO(NUSE,N1)	PREP0497
		N E = NE - NU SE	PREP0498
		IDGE(IR, JR) = IDGE(IR, JR) + NUSE	PREP0499
		IDGE (JR, IR) = IDGE (JR, IR) + NUSE	PREP0500
		IDGE(JR, 20) = IDGE(JR, 20) + NUSE	PR EP 0 50 1
		IDGE(IR,20) = IDGE(IR,20) + NUSE	PREP0502
		CONTINUE	PREP0503
	350	CONTINUE	PREP0504

	CALL BUGOUT (8HIDGE-H , IDGE, 1, NR, 15, 1, NR, 20, 1)	PREP0505
	CALL BUGOUT (8HIDGE-H , IDGE, 1, NR, 15, 16, 20, 20, 1)	PREP0506
	NSUM=0	PREP0507
	DO 355 IR=NRM, NR	PREP0508
	K=4*NAREG(IR, IC) - IDGE(IR, 20)	PREP0509
	IF $(K.EQ.U)$ GO TO 355	PREP0510
	N SUM=NSUM+K	PREP0511
	NR2=IR	PREP0512
355	CONTINUE	PREP0513
333	NK1=NR2-1	PREPO514
	DO 380 IK = NRM, NR1	PREP0515
	NE=4*NAREG(IR, IC) - IDGE(IR, 20)	PREP0515
	A = NE	PREPOSIO
	IF (NSUM-NE-LE-0) GO TO 380	PREPOS18
	A = A / (NSUM - NE)	PREP0519
	KR = IR + 1	PREP0520
	DO $370 \text{ JR}=KR, NR2$	PKEP0521
	K=4*NAREG(JR, IC) - IDGE(JR, 20)	PREP0522
	NUSE=A*K+0.5	PREP0523
	NUSE=MINO(NUSE,NE,K)	PREPO524
	NE=NE-NUSE	PREP0525
	NSUM=NSUM-2*NE	PREP0526
	IDGE (IR, JR) = IDGE (IR, JR) + NUSE	PREP0527
	1 DGE(1R, 20) = 1 DGE(1R, 20) + NUSE	PREP0528
	IDGE (JR, 20) = IDGE (JR, 20) + NUSE	PREP0529
	IDGE (JR, IR) = IDGE (JR, IR) + NUSE	PREP0530
370	CONTINUE	PREP0531
	CONTINUE	PREP0532
300	CALL BUGOUT (8HIDGE-I , IDGE, 1, NR, 15, 1, NR, 20, 1)	PREP0533
	CALL BUGOUT (8HIDGE-I , IDGE, 1, NR, 15, 16, 20, 20, 1)	PREP0534
	DO 390 $I R = 1, NR$	PREP0535
	K=4*NAREG(IR,IC)-IDGE(IR,20)	PREP0535
	IDGE (IR, IR) = IDGE (IR, IR) + K	PREPOSSO PREPOSS7
39.0	CONTINUE	PREP0537 PREP0538
570	CALL BUGOUT (8HIDGE-J , IDGE, 1, NR, 15, 1, NR, 20, 1)	PREP0539
	CALL BUGOUT (8HIDGE-J , IDGE, 1, NR, 15, 16, 20, 20, 1)	PREP0539 PREP0540
		PREPUSAU

	PREP0541
PACK 1DGE INTO NEDGE ARRAY	PREPOSAT PREPOSA2
CONTINUE	PREP0543
N=O	PREPOS43
CALL BUGOUT (8HCYCLE , IC, 1, 1, 1, 1, 1, 1, 1)	PREPO544 PREP0545
CALL BUGOUT (8HIDGE , IDGE, 1, NR, 15, 1, 20, 20, 1)	PKEP0545
DO 460 $J=1,NR$	PREPOSA6 PREPOSA7
DO 460 $J_{J}=J_{1}$ 16	
•	PREP0548
IF $(IDGE(J,JJ)-GT-0)$ N=N+1 CONTINUE	PREP0549
	PREP0550
I = 0	PREP0551
$\frac{1F}{1N2} = \frac{IC}{F} \frac{IC}{IC}$	PREP0552
IN2=IEDGE(IC)	PREP0553
1N3 = IEDGE (21) $IR (IN2 = IN CR N) = CR MR (180)$	PREP0554
IF (IN2-IN.GE.N) GO TO 480	PREP0555
N=N-(1N2-1N)	PREP0556
IF (IN2.EQ.IN3) GO TO 470	PREP0557
IN4=IN2+1	
DO $465 J = IN4, IN3$	PREP0559 17
K=IN3-J+IN4	PREP0560
NEDGE $(1, K+N) = NEDGE (1, K)$	PREP0561
NEDGE $(2, K+N) = NEDGE (2, K)$	PREP0562
CONTINUE	PREP0563
CONTINUE DO 425 JEIC 21	PREP0564
DO 475 $J=IC,21$	PREP0565
IEDGE (J) = IEDGE (J) + N	PREP0566
CONTINUE	PREP0567
$\mathbf{N} = 0$	PREP0568
JO 490 J=1, NR	PREP0569
DO 490 JJ=J,16	PREP0570
IF (IDGE (J,JJ) .EQ.0) GO TO 490	PREP0571
IN=IN+1	PREP0572
K=JJ	PREP0573
IF $(JJ.EQ.16)$ K=0	PREP0574
NEDGE $(1, IN) = 100 * K + J$	PREP0575
NEDGE(2,IN) = IDGE(J,JJ)	PREP0576

			000000 577
		CONTINUE	PREP0577 PREP0578
С	500	CONTINUE	PREPOS78
L		SET UP NUMBER OF ASM DISCHARGED EACH CYCLE	PREP0580
		DO 510 IC=1,21	
	E 4 0	DO 510 IR=1,15	PREP0581
	510	NADIS $(IR, IC) = 0$	PREP0582
		DO 530 IC=1,20	PREP0583
		IF (ISURCH(IC).EQ.0) GO TO 530	PREP0584
		NR=NREG (IC)	PREP0585
		IC1=IC+1	PREP0586
		DO 525 IR=1, NR	PREP0587
		NADIS $(IR, IC) = NAREG(IR, IC)$	PREP0588
		IF (IC.EQ.20) GO TO 525	PREP0589
		K=100*IC+IR	PREP 0590
		DO 520 JC=IC1,20	PREP0591
		IF (ISURCH (JC) . EQ. 0) GO TO 520	PREP0592
		NR 1 = NREG (JC)	PREP0593
		DO 515 JR=1, NR1	PREP0594
		IF (JSHUF(JR,JC)-NE-K) GO TO 515	PREP0595
		NADIS (IR, IC) = NADIS (IR, IC) - NAREG (JR, JC)	PREP0596
		CONTINUE	PREP0597
	520	CONTINUE	PREP0598
		IF (NADIS(IR,IC).LT.O) IERR=1	PREP0599
		IF (NADIS(IR,IC).LT.O) WRITE (106,902) IR,IC	PREP0600
	525	CONTINUE	PREP0601
		CALL BUGOUT (8HNADIS , NADIS, 1, NR, 15, IC, IC, 21, 1)	PREP0602
	530	CONTINUE	PREP0603
С		EQUILLIBRIUM CYCLE	PREP0604
		NR=NREG (21)	PREP0605
		IF (ISURCH (21) . EQ. 0) GO TO 545	PREP0606
		DO 540 IH=1,NR	PREP0607
		NADIS $(IR, 21) = NAREG (IR, 21)$	PREP0608
		K=2100+1R	PREP0609
		DO 535 $JR=1$, NR	PREP0610
		IF (JSHUF (JR, 21).NE.K) GO TO 535	PREP0611
		NADIS $(IR, 21) = NADIS (IR, 21) - NAREG (JR, 21)$	PREP0612

	535 CONTINUE	PREP0613
	540 CONTINUE	PREPO614
	CALL BUGOUT (8HNADIS , NADIS, 1, NR, 15, 21, 21, 21, 1)	PREP0615
	545 CONTINUE	PREP0616
С		PREP0617
С	GENERATE ISOTOPIC DATA FROM FIT	PREP0618
С		PREP0619
	IF (ISO.NE1) GO TO 570	PREP0620
	150=0	PREP0621
	DO 565 $I=1$, NRICH	PR EP 06 22
	E = ENRICH(I)	PREP0623
	DO 560 J=1, NBU	PREP0624
	B = BURNUP(J)	PREP0625
	ISOTOP $(J, I, 1) = EXP(-0.00159*B*(1.0-4.1E-3*B))$	PREP0626
	ISOTOP (J, I, 2) = E*EXP (-0.1162*B*(1.0+2.75E-5*B*B*E) /E**0.965)	PREP0627
	ISOTOP (J, I, 3) =4.795*E**0.40* (1.0-EXP(1425*B/E**0.6))	PREP0628
	560 CONTINUE	PREP0629
	565 CONTINUE	PREP0630
	570 CONTINUE	PREP0631
	IF (IERR.EQ.1) CALL EXIT	PREP0632
	RETURN	PREP0633
С		PREP0634
	901 FORMAT (1H0,9X,48H*** ERROR, CAN NOT GENERATE EDGE DATA FOR A CORI	E PREP0635
	1 6H WITH , I3, 16H ASSEMBLIES AND , I2, 18H ASSEMBLIES ACROSS)	
	902 FORMAT (1H0,9X,47HERROR, NEGATIVE NUMBER OF ASSEMBLIES DISCHARGED	
	1 12H FROM REGION, I3, 7H, CYCLE, I3 /)	PREP0638
С		PREP0639
	END	PREP0640

SUBROUTINE INEDIT	INED0001
	INED0002
SUBROUTINE INEDIT EDITS OUT BASIC INPUT DATA	INED0003
	INED0004
COMMON /TABL1/ NBU, NRICH, BURNUP (20), ENRICH (20), ITYPE (20), BIAS (20),	INEL0005
1 BPWRTH (20), CYBIAS	INED0006
COMMON /IO/ IO5,IO6,IO7	INED0007
COMMON /TABL2/ KINF (20,20), ISOTOP (20,20,6), POWFIS (20)	INED0008
REAL KINF, ISOTOP	INED0009
COMMON /BASIC/ TITLE(20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT,	INEDO010
1 BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG	INED0011
LOGICAL DEBUG	INED0012
COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21),	INED0013
1 NFEED (21) , NREG (21) , CYLOAD (21) , CAPFAC (21) , CYTIM (21) ,	INED0014
2 SDLEN (21), IFEDGE (21)	INED0015
COMMON /RGDAT/ NEDGE(2,1000), IEDGE(21), NAREG(15,21), JSHUF(15,21),	INED0016
1 BPFRAC (15,21), BOCBU (15,21), EOCBU (15,21),	INED0017
2 IDXREG (15,21), NADIS (15,21)	INED0018
COMMON /CYCL1/ C1RICH(10),C1TYPE(10),C1ID(10),C1LOAD(10),NAC1(10)	INED0019
COMMON / FEED / NAFEED (21, 3), FTYPE (21, 3), FLOAD (21, 3), FRICH (21, 3),	INED0020
1 FID(21,3)	INED0021
INTEGER FTYPE, C1TYPE	INED0022
COMMON /COST / IFCOST	INED0023
COMMON /COEF / COFIT(11),TEMP(2),BWRTH,DPWRTH,XEWRTH	INED0024
	INED0025
DIMENSION ISONAM (3)	INED0026
DATA NCASE / 0 /	INED0027
DATA ISONAM / 4H U , 4HU-35, 4H PU /	INED0028
•	INED0029
NCASE=NCASE+1	INED0030
CALL HEAD	INED0031
WRITE (106,101) NASS, NADIAM, HEIGHT, ULOAD, POWER	INED0032
WRITE (106, 102) W, ALPHA, ELIM, BUDIF, CYBIAS	INED0033
WRITE (106,114) BWRTH, DPWRTH, XEWRTH	INED0034
IF (1HAL.GT.0) WRITE (106,103)	INED0035
IF (IFCOST.GT.0) WRITE (106,104)	INED0036

C C C

С

С

	T UND 00 37
IF (NCASE.GT.1) GO TO 25	INED0037
DO 12 N1=1, NRICH, 8	INED0038
N2=MINO(NRICH,N1+7)	INED0039
CALL HEAD	INED0040
WRITE (106,105)	INELOO41
WRITE $(106, 106)$ (ENRICH (I) , $I = N1, N2$)	INEDO042
WRITE (106,107) (ITYPE(I), I=N1, N2)	INED0043
WRITE (106, 108) (POWFIS(I), I=N1, N2)	INEDOO44
WRITE (106,109)	INED0045
DO 10 I=1, NBU	INED0046
WRITE (IO6, 110) BURNUP(I), BIAS(I), BPWRTH(I), (KINF(I,J), J=N1, N2)	INED0047
10 CONTINUE	INED0048
12 CONTINUE	INED0049
NISO=3	INED0050
N1=1	INED0051
N 2=NR ICH	INED0052
DO 20 K=1, NISO	INED0053
CALL HEAD	INED0054
WRITE (106,111) ISONAM(K)	INED0055
WRITE (106, 112) (ENRICH(I), I=N1, N2)	INED0056
DO 15 I=1, NBU	INED0057
WRITE (IO6, 113) BURNUP(I), (ISOTOP(I, J, K), $J=N1, N2$)	INED0058
15 CONTINUE	INED0059
20 CONTINUE	INED0060
25 CONTINUE	INED0061
RETURN	INED0062
C	INED0063
101 FORMAT (1HO, 9X,24HNUMBER OF ASSEMBLIES, 16 /	INED0064
1 10X,24HASSEMBLIES ACROSS CORE 16 /	INED0065
2 10 X, 24 HCORE HEIGHT	INED0066
3 10X,24HAVERAGE CORE LOADING, F6.2,4H MTM /	IN ED0067
4 10X,24HCORE THERMAL POWER, F6.0,4H MW)	INED0068
102 FORMAT(1H0, 9X,24HPROB. OF ABSORP. (W),F6.3 /	INEDOO69
$1 \qquad 10 X_2 24 HALBEDO F6.3 /$	INED0070
2 10X,24HMINIMUM ENRICHMENT,F6.3, 4H W/0 /	INED0071
3 10X,24HMAXIMUM ENRICHMENT, F6.3, 4H W/0 /	
J IUX/24MIXX101 ENALCHIENT	20020072

`

. .

5 10x,24HCycle B1AS FACTOR,F6.3) INEDO 103 FORMAT (1H0,9X,33HA HALING CALCULATION WILL BE USED) INEDO 104 FORMAT (1H0,9X,32HFUEL COST DATA WILL BE GENERATED) INEDO 105 FORMAT (1H0,9X,32HFUEL COST DATA WILL BE GENERATED) INEDO 105 FORMAT (38X,34HK EFF VERSUS BURNUP AND ENRICHMENT //) INEDO 106 FORMAT (10x, 10HENRICHMENT, 14X,8F10.3) INEDO 107 FORMAT (10x, 10HFUEL TYPE , 11X,8I 10) INEDO 108 FORMAT (10x, 10HFUEL TYPE , 11X,8I 10) INEDO 109 FORMAT (140,9X,6HBURNUP,3X,4HBIAS,4X,7HBP WETH,15X,5HK EFF /) INEDO 110 FORMAT (8X,P8.3,F7.4,P11.2,8F10.5) INEDO 111 FORMAT (30X,15HISOTOPIC DATA (,A4,30H) VERSUS BURNUP AND ENRICHMENT INEDO 112 FORMAT (18X,15F8.3) INEDO 113 FORMAT (140, 9X,24HBORON WORTH	(1		INED0073
103 FOR MAT (1H0,9X,33HA HALING CALCULATION WILL BE USED) INEDO 104 FORMAT (1H0,9X,32HFUEL COST DATA WILL BE GENERATED) INEDO 105 FORMAT (1H0,9X,32HFUEL COST DATA WILL BE GENERATED) INEDO 105 FORMAT (1H0,9X,32HFUEL COST DATA WILL BE GENERATED) INEDO 105 FORMAT (1H0,9X,32HFUEL COST DATA WILL BE GENERATED) INEDO 106 FORMAT (10X,10HENRICHMENT,14X,8F10.3) INEDO 107 FORMAT (10X,10HFUEL TYPE , 11X,8I10) INEDO 108 FORMAT (10X,10HFUEL TYPE , 11X,8F10.3) INEDO 109 FORMAT (1H0,9X,6HBURNUP,3X,4HBIAS,4X,7HBP WBTH,15X,5HK EFF /) INEDO 110 FORMAT (30X,15HISOTOPIC DATA (,A4,30H) VERSUS BURNUP AND ENRICHMENT INEDO 111 FORMAT (18X,15F8.3) INEDO 112 FORMAT (10X,F8.3,15F8.4) INEDO 113 FORMAT (1H0, 9X,24HBORON WORTH	4	10X,24HBURNUP VARIATION PARAM, F6. 3,8H GWD/MTM /	INED0074
104 FORMAT(1H0,9X,32HFUEL COST DATA WILL BE GENERATED) INEDO 105 FORMAT(38X,32HFUEL COST DATA WILL BE GENERATED) INEDO 106 FORMAT(38X,32HFUEL COST DATA WILL BE GENERATED) INEDO 106 FORMAT(38X,32HFUEL COST DATA WILL BE GENERATED) INEDO 106 FORMAT(38X,32HFUEL COST DATA WILL BE GENERATED) INEDO 106 FORMAT(10X,10HENRICHMENT,14X,8F10.3) INEDO 107 FORMAT(10X,10HFUEL TYPE ,11X,8I10) INEDO 108 FORMAT(10X,10HFUEL TYPE ,11X,8F10.3) INEDO 109 FORMAT(1H0,9X,6HBURNUP,3X,4HBIAS,4X,7HBP WBTH,15X,5HK EFF /) INEDO 110 FORMAT(30X,15HISOTOPIC DATA (,A4,30H) VERSUS BURNUP AND ENRICHMENT INEDO 111 FORMAT(18X,15HISOTOPIC DATA (,A4,30H) VERSUS BURNUP AND ENRICHMENT INEDO 112 FORMAT(18X,15F8.3) INEDO INEDO 112 FORMAT(10X,F8.3,15F8.4) INEDO INEDO 114 FORMAT(1H0, 9X,24HBORON WORTH	2	•	
104 FORMAT (1H0,9X,32HFUEL COST DATA WILL BE GENERATED) INEDO 105 FORMAT (38X,34HK EFF VERSUS BURNUP AND ENRICHMENT //) INEDO 106 FORMAT (10X, 10HENRICHMENT, 14X,8F10.3) INEDO 107 FORMAT (10X, 10HFUEL TYPE , 11X,8I10) INEDO 108 FORMAT (10X, 10HFUEL TYPE , 11X,8F10.3) INEDO 109 FORMAT (10X, 10HPOWER/FIS., 14X,8F10.3) INEDO 109 FORMAT (100,9X,6HBURNUP,3X,4HBIAS,4X,7HBP WBTH,15X,5HK EFF /) INEDO 110 FORMAT (30X,15HISOTOPIC DATA (,A4,30H) VERSUS BURNUP AND ENRICHMENT INEDO 1 //) INEDO 112 FORMAT (18X,15F8.3) INEDO 113 FORMAT (10X,F8.3,15F8.4) INEDO 114 FORMAT (1H0, 9X,24HBORON WORTH,F6.1,8H PCM/PPM / INEDO 2 10X,24HDOPPLER DEFECT,F6.0,4H PCM / INEDO 2 10X,24HXENON DEFECT,F6.0,4H PCM / INEDO	103 FORMAT (FORMAT (1H0,9X,33HA HALING CALCULATION WILL BE USED)	INED0075
105 FORMAT (38X, 34HK EFF VERSUS BURNUP AND ENRICHMENT //) INEDO 106 FORMAT (10X, 10HENRICHMENT, 14X, 8F10.3) INEDO 107 FORMAT (10X, 10HFUEL TYPE, 11X, 8I10) INEDO 108 FORMAT (10X, 10HFUEL TYPE, 11X, 8F10.3) INEDO 109 FORMAT (10X, 10HPOWER/FIS., 14X, 8F10.3) INEDO 109 FORMAT (140, 9X, 6HBURNUP, 3X, 4HBIAS, 4X, 7HBP WRTH, 15X, 5HK EFF /) INEDO 110 FORMAT (8X, F8.3, F7.4, F11.2, 8F10.5) INEDO 111 FORMAT (30X, 15HISOTOPIC DATA (,A4, 30H) VERSUS BURNUP AND ENRICHMENT INEDO 1 //) INEDO INEDO 112 FORMAT (18X, 15F8.3) INEDO 113 FORMAT (10X, F8.3, 15F8.4) INEDO 114 FORMAT (1HO, 9X, 24HBORON WORTH, F6.1, 8H PCM/PPM / INEDO 2 10X, 24HBORON WORTH, F6.0, 4H PCM / INEDO 2 10X, 24HXENON DEFECT, F6.0, 4H PCM / INEDO			INED0076
107 FORMAT (10X, 10HFUEL TYPE, 11X, 8110) INEDO 108 FORMAT (10X, 10HPOWER/FIS., 14X, 8P10.3) INEDO 109 FORMAT (1H0, 9X, 6HBURNUP, 3X, 4HBIAS, 4X, 7HBP WRTH, 15X, 5HK EFF /) INEDO 110 FORMAT (1H0, 9X, 6HBURNUP, 3X, 4HBIAS, 4X, 7HBP WRTH, 15X, 5HK EFF /) INEDO 110 FORMAT (30X, 15H1SOTOPIC DATA (,A4, 30H) VERSUS BURNUP AND ENRICHMENT INEDO 111 FORMAT (18X, 15H1SOTOPIC DATA (,A4, 30H) VERSUS BURNUP AND ENRICHMENT INEDO 112 FORMAT (18X, 15F8.3) INEDO INEDO 113 FORMAT (10X,F8.3, 15F8.4) INEDO 114 FORMAT (1HO, 9X, 24HBORON WORTH ,F6.1,8H PCM/PPM / INEDO 2 10X, 24HDOPPLER DEFECT ,F6.0,4H PCM / INEDO 2 10X, 24HXENON DEFECT ,F6.0,4H PCM / INEDO			INED0077
108 FORMAT (10X, 10HPOWER/FIS., 14X, 8P10.3) INEDO 109 FORMAT (140, 9X, 6HBURNUP, 3X, 4HBIAS, 4X, 7HBP WRTH, 15X, 5HK EFF /) INEDO 110 FORMAT (8X, F8.3, F7.4, F11.2, 8F10.5) INEDO 111 FORMAT (30X, 15H1SOTOPIC DATA (,A4, 30H) VERSUS BUBNUP AND ENRICHMENT INEDO 1 //) INEDO 112 FORMAT (18X, 15F8.3) INEDO 113 FORMAT (10X,F8.3, 15F8.4) INEDO 114 FORMAT (1HO, 9X,24HBORON WORTH	106 FURMAT (PURMAT (10X, 10HENRICHMENT, 14X, 8F10.3)	INED0078
109 FORMAT (1H0,9X,6HBURNUP,3X,4HBIAS,4X,7HBP WRTH,15X,5HK EFF /) INEDO 110 FORMAT (8X,F8.3,F7.4,F11.2,8F10.5) INEDO 111 FORMAT (30X,15H1SOTOPIC DATA (,A4,30H) VERSUS BURNUP AND ENRICHMENT INEDO 1 //) INEDO 112 FORMAT (18X,15F8.3) INEDO 113 FORMAT (10X,F8.3,15F8.4) INEDO 114 FORMAT (1H0, 9X,24HBORON WORTH ,F6.1,8H PCM/PPM / INEDO 2 10X,24HDOPPLER DEFECT ,F6.0,4H PCM / INEDO 2 10X,24HXENON DEFECT ,F6.0,4H PCM / INEDO			INED0079
109 FORMAT (1H0,9X,6HBURNUP,3X,4HBIAS,4X,7HBP WRTH,15X,5HK EFF /) INEDO 110 FORMAT (8X,F8.3,F7.4,F11.2,8F10.5) INEDO 111 FORMAT (30X,15HISOTOPIC DATA (,A4,30H) VERSUS BURNUP AND ENRICHMENT INEDO 1 //) INEDO 112 FORMAT (18X, 15F8.3) INEDO 113 FORMAT (10X,F8.3, 15F8.4) INEDO 114 FORMAT (1H0, 9X,24HBORON WORTH F6.1,8H PCM/PPM / 2 10X,24HDOPPLER DEFECT F6.0,4H PCM / 2 10X,24HXENON DEFECT F6.0,4H PCM /	108 FORMAT	ORMAT (10X, 10HPOWER/FIS., 14X, 8P10.3)	INED0080
110 FORMAT (8X, F8.3, F7.4, F11.2, 8F10.5) INEDO 111 FORMAT (30X, 15H1SOTOPIC DATA (,A4,30H) VERSUS BURNUP AND ENRICHMENT INEDO 1 //) INEDO 1 //) INEDO 112 FORMAT (18X, 15F8.3) INEDO 113 FORMAT (10X, F8.3, 15F8.4) INEDO 114 FORMAT (1HO, 9X, 24HBORON WORTH F6.1, 8H PCM/PPM / INEDO 2 10X, 24HDOPPLER DEFECT F6.0, 4H PCM / INEDO 2 10X, 24HXENON DEFECT F6.0, 4H PCM / INEDO			INED0081
1 //) INEDO 112 FORMAT (18X, 15F8.3) INEDO 113 FORMAT (10X, F8.3, 15F8.4) INEDO 114 FORMAT (1HO, 9X, 24HBORON WORTH	110 FORMAT (8	PORMAT (8X, F8.3, F7.4, F11.2, 8F10.5)	INED0082
112 FORMAT (18X, 15F8.3) INEDO 113 FORMAT (10X, F8.3, 15F8.4) INEDO 114 FORMAT (1HO, 9X, 24HBORON WORTH	111 FORMAT	ORMAT (30X, 15HISOTOPIC DATA (,A4, 30H) VERSUS BURNUP AND ENRICHMENT	INED0083
113 FORMAT (10X, F8. 3, 15F8. 4) INEDO 114 FORMAT (1HO, 9X, 24HBORON WORTH, F6. 1,8H PCM/PPM / INEDO 2 10X, 24HDOPPLER DEFECT, F6. 0,4H PCM / INEDO 2 10X, 24HXENON DEFECT, F6. 0,4H PCM / INEDO	1 /	//)	INED0084
114FORMAT (1HO, 9X, 24HBORON WORTH, F6.1,8H PCM/PPM /INEDO210X, 24HDOPPLER DEFECT, F6.0,4H PCM /INEDO210X, 24HXENON DEFECT, F6.0,4H PCM)INEDO	112 FORMAT (ORMAT (18X, 15F8.3)	INED0085
210X,24HDOPPLER DEFECT	113 FORMAT (ORMAT (10X,F8.3,15F8.4)	INED0086
2 10X, 24 HXENON DEFECT	114 FORMAT	FORMAT (1HO, 9X, 24HBORON WORTH	INED0087
	2	10X,24HDOPPLER DEFECT	INED0088
	2	10X,24HXENON DEPECT	INED0089
CINEDO			INED0090
END	END	SND	INED0091

SUBROUTINE CALC	CALC0001
	CALCO002
CALCULATE CYCLE BY CYCLE DATA	CALC0003
	CALC0004
COMMON /TABL1/ NBU, NRICH, BURNUP(20), ENRICH(20), ITYPE(20), BIAS(20),	CALC0005
1 BPWRTH (20) , CYBIAS	CALC0006
COMMON /10/ 105,106,107	CA LC 0007
COMMON /TABL2/ KINF (20,20), ISOTOP (20,20,6), POWFIS (20)	CALC0008
REAL KINF, ISOTOP	CALC0009
COMMON /BASIC/ TITLE(20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT,	CALC0010
1 BUDIF, ISO, W, ALPHA, ELIM(2), IHAL, POWER, DEBUG	CALC0011
LOGICAL DEBUG	CALC0012
COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21),	CALCO013
1 NFEED (21), NREG (21), CYLOAD (21), CAPFAC (21), CYTIM (21),	CALC0014
2 SDLEN (21), IFEDGE (21)	CALCO015
COMMON /RGDAT/ NEDGE(2,1000), IEDGE(21), NAREG(15,21), JSHUF(15,21),	CALC0016
1 BPFRAC (15,21), BOCBU (15,21), EOCBU (15,21),	CALC0017
2 IDXREG (15,21), NADIS (15,21)	CALC0018
COMMON /CYCL1/ C1HICH (10), C1TYPE (10), C1ID (10), C1LOAD (10), NAC1 (10)	CALC0019
COMMON /FEED / NAFEED (21, 3), FTYPE (21, 3), FLOAD (21, 3), FRICH (21, 3),	CALC0020
1 FID(21,3)	CALC0021
INTEGER FTYPE, C1TYPE	CALC0022
COMMON /COST / IFCOST	CALC0023
COMMON /COEF / COFIT (11), TEMP (2), BWRTH, DPWRTH, XEWRTH	CALC0024
COMMON /EDGE/ BFRAC (15,20)	CALC 0025
COMMON /CYDEP/ BSHARE (15, 10), BBU (10), BKEFF (10), NBBU	CALC0026
	CALC0027
DIMENSION REGE(15), IRTYPE(15), REGBU(15), RLOAD(15)	CALCO028
	CALC0029
DO 100 $IC=1,20$	CALC0030
IF (ISURCH(IC).EQ.0) GO TO 100	CA LC 00 3 1
IN1=1	CALC0032
IF (IC.GT.1) IN1=IEDGE(IC-1)+1	CALC0033
IN2 = IEDGE(IC)	CALC0034
DO 5 $I=1, 16$	CALC0035
DO 5 J=1, 15	CALC0036

C C C

С

С

5	RFRAC(J,I) = 0.0	CALC0037
-	DU 15 I=IN1,IN2	CALC0038
	K = N EDG E(1, I)	CALC0039
	A = N E D G E (2, I) * 0.25	CALC0040
	JR=K/100	CALCOO41
	IR=MOD(K,100)	CA LC0042
	IF (K.EQ.0) GO TO 15	CALC 0043
	IF (JR.EQ.0) GO TO 10	CALC0044
	RFRAC $(JK, IR) = A / NAREG (JR, IC)$	CALC0045
	RFRAC(IR, JR) = A/NAREG(IR, IC)	CALCO046
	GO TO 15	CALC0047
10	CONTINUE	CALCO048
	RFRAC (IR, 16) = A/NAKEG (IR, IC)	CALCO049
15	CONTINUE	CALC0050
	NR=NREG (IC)	CALC0051
	SUM=0.0	CALCU052
	DO 30 IR=1,NR	CALC0053
	REGBU(1R) = BOCBU(IR, 1C)	CALC0054
	K=IDXREG(IR,IC)	CALCO055
	JR=MOD(K,100)	CALC0056
	JC=K/100	CALCO057
	IF (JC.EQ.1) GO TO 20	CALC0058
	R EG E (IR) = FRICH (JC, JR)	CALC0059
	IRTYPE (IR)=FTYPE (JC, JR)	CALC0060
	k LOAD(IR) = FLOAD(JC, JR)	CALC0061
	SUM=SUM+RLOAD (IR) *NAREG (IR, IC)	CALCOU62
	GO TO 30	CALC0063
20	CONTINUE	CALCO064
	REGE (IR) = C1RICH (JR)	CALC0065
	IRTYPE(IR) = C1TYPE(JR)	CALC0066
	RLOAD (IR) = C1LOAD (JR)	CALC0067
	SUM=SUM+RLOAD (IR) *NAREG (IR, IC)	CALC0068
30	CONTINUE	CALC0069
	CYLOAD (IC) = SUM	CALC0070
	IF $(CYBU(IC) \cdot EQ. 0.0)$ CYBU $(IC) = 1 \cdot E - 5 * CAPFAC(IC) * CYTIM(IC) * POWER/$	CALC0071
	1 CYLOAD (IC)	CALC0072

~

		011 00073
	SUM=ANASS/SUM	CALC0073
	DO 35 IR=1,NR	CALC0074
35	R LOAD (IR) = RLOAD (IR) * SUM	CALC 0075
	IS=ISURCH(IC)	CALCOO76
	IF (IS.LE.10) GO TO 45	CALC0077
	K=100*IC+IS-10	CALC0078
	DO 40 IR=1, NR	CALC0079
	IF (IDXREG(IR, TC) . EQ.K) IS=10+IR	CALC0080
40	CONTINUE	CALC0081
	CONTINUE	CALC0082
	CALL CYCALC (IS, CYBU (IC), EOLK (IC), BP (IC), NR, REGBU, REGE, IRTYPE,	CALCO083
	1 NAREG (1, IC), RFRAC, BPFRAC (1, IC), RLOAD)	CALC 0084
	IF (IS.LE. 10) GO TO 50	CALC0085
	JS=ISURCH(IC)-10	CALC0086
	FRICH (IC, JS) = REGE (IS-10)	CALCO087
50	CONTINUE	CALC0088
	DO 75 IR=1, NR	CALC0089
	EOCBU (IR, IC) = REGBU (IR)	CALC0090
	FRAC=NADIS(IR, IC)	CALC0091
	FRAC=FRAC/NAREG(IR, IC)	CALC0092
	K=100 * IC + IR	CALC0093
	IF (1C. EQ. 20) GO TO 75	CALCO094
	IC1=IC+1	CALCO095
	DO 60 JC=IC1,20	CALCO096
	NR1 = NREG (JC)	CALC0097
	IF (ISURCH(JC).EQ.0) GO TO 60	CALC0098
		 CALC0099
	IF (JSHUF (JR, JC).NE.K) GO TO 55	CALCO 100
	IF (BOCBU (JR, JC).GT.0) GO TO 55	CALCO101
	BOCBU (JR, JC) = EOCBU (IR, IC) - BU DIF * FRAC	CALCOIDT
55	CONTINUE	CALCOIO2 CALCOIO3
		CALCO104
10	CONTINUE	CALC0105
100	CALL CYEDIT (IC, REGE, IRTYPE)	CALC0106
100	CONTINUE	CALC0107
		CALC0108

RETURN END •

CALCO109 CALCO110

~

SUBROUTINE EQCALC		0001
		0002
CALCULATE EQUILIBRIUM CYCLE	-	0003
		0004
COMMON /TABL1/ NBU, NRICH, BURNUP (20), ENRICH (20), ITYPE (20), BIAS (20),		0005
1 BPWRTH (20), CYBIAS	EQC	0006
COMMON /IO/ I05,I06,I07	EQC	0007
COMMON /TABL2/ KINF(20,20), ISOTOP(20,20,6), POWFIS(20)	EQC	0008
REAL KINF, ISOTOP	EQC	0009
COMMON /BASIC/ TITLE (20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT,	EQC	0010
1 BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG	EQC	0011
LOGICAL DEBUG		0012
COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21),	EQC	0013
1 NFEED (21), NREG (21), CYLOAD (21), CAPFAC (21), CYTIM (21),	_	0014
2 SDLEN (21), IFEDGE (21)		0015
COMMON /RGDAT/ NEDGE(2,1000), IEDGE(21), NAREG(15,21), JSHUF(15,21),		0016
1 $BPFRAC(15, 21), BOCBU(15, 21), EOCBU(15, 21),$	-	0017
2 I DXR EG (15, 21), NADIS (15, 21)		0018
COMMON /CYCL1/ C1RICH(10), C1TYPE(10), C1ID(10), C1LOAD(10), NAC1(10)	-	0019
COMMON /FEED / NAFEED (21, 3), FTYPE (21, 3), FLOAD (21, 3), FRICH (21, 3),		0020
1 FID(21,3)		0021
INTEGER FTYPE, C1TYPE	-	0022
COMMON /COST / IFCOST	-	0023
COMMON /COEF / COFIT(11), TEMP(2), BWRTH, DPWRTH, XEWRTH	-	0024
COMMON /EDGE/ RFRAC (15,20)		0025
	-	0026
DIMENSION REGE(15), RTYPE(15), REGBU(15), PSHAR(15), RLOAD(15)		0027
DIMENSION INDEX (15)	-	0028
INTEGER HTYPE	-	0029
		0030
IS=ISURCH(21)		0031
IF (IS.EQ.O) RETURN	-	0032
NR = NREG(21)		
NF = NF ELD(21)		0033
ITER=0		0034
DO 5 I=1, NR		0035
	EQC	0036

.

C C C

С

C

~		n o o	0007
5	PSHAR(I) = 1.0		0037 0038
	IN1=IEDGE(20)+1		
	IN2=IEDGE(21)		0039
	DO 205 I=1,16	-	0040
	DO 205 J=1,15		0041
	$\mathrm{HFRAC}\left(\mathbf{J},\mathbf{I}\right)=0.0$		0042
10	CONTINUE	-	0043
	ITER=ITER+1		0044
	IF (ITER.GT.50) GO TO 400		0045
	JITER=0	EQC	0046
15	CONTINUE	EQC	0047
	JITER=JITER+1	EQC	0048
	IF (JITER.GT.50) GO TO 400	EQC	0049
20	CONTINUE	EQC	0050
	S UM = 0.0	EQC	0051
	DO 25 JR=1, NR	EQC	0052
25	INDEX(JR)=0	EQC	0053
	N N=O	EQC	0054
30	CONTINUE	EQC	0055
	DO 50 JR=1, NR	EQC	0056
	IF (INDEX(JR).GT.0) GO TO 50	EQC	0057
	IF (JSHUF (JR, 21).GE. 100) GO TO 35	EQC	0058
	JF=JSHUF(JR, 21)	EQC	0059
	N N = N N + 1	EQC	0060
	INDEX (JR) = 1	EQC	0061
	RLOAD(JR) = FLOAD(21, JF)	EQC	0062
	REGE(JR) = FRICH(21, JF)	EQC	0063
	SUM=SUM+RLOAD (JR) *NAREG (JR, 21)	EQC	0064
	RTYPE(JR) = FTYPE(21, JF)	EQC	0065
	REGBU(JR) = 0.0	EQC	0066
	BOCBU $(JR, 21) = 0.0$	-	0067
	EOCBU $(JR, 21) = PSHAR (JR) * CYBU (21)$		0068
	GO TO 50		0069
35	CONTINUE		0070
_	JK=JSHUF(JR,21)	-	0071
	N = MOD (JK, 100)		0072

	IF (INDEX(N).EQ.0) GO TO 50	EQC 0073
	KR=JR	EQC 0074
	N N = N N + 1	EQC 0075
	INDEX (JR) = 1	EQC 0076
	FRAC = NADIS(N, 21)	EQC 0077
	FRAC=FRAC/NAREG(N,21)	EQC 0078
	REGBU (KR) =CYBU (21) *PSHAR (N) +REGBU (N) - BUDIF*FRAC	EQC 0079
	BOCBU (KR, 21) = $R E G B U$ (KR)	EQC 0080
	RTYPE(KR) = RTYPE(N)	EQC 0081
	EOCBU (KR, 21) = REGBU (KR) + CYBU (21) * PSHAR (KR)	EQC 0082
	REGE (KR) = REGE (N)	EQC 0083
	RLOAD(KR) = RLOAD(N)	EQC 0084
	SUM=SUM+ALOAD (KR) *NAREG (KR,21)	EQC 0085
50	CONTINUE	EQC 0086
	IF (NN.LT.NR) GO TO 30	EQC 0087
	CYLOAD(21) = SUM	EQC 0088
	SUM=ANASS/SUM	EQC 0089
	DO 55 $JR=1$, NR	EQC 0090
	RFRAC(JR, JR) = 0.0	EQC 0091
55	RLOAD (JR) = SUM * RLOAD (JR)	EQC 0092
	DO 215 I=IN1,IN2	EQC 0093
	K=NEDGE (1,I)	EQC 0094
	A = NEDGE(2, I) * 0.25	EQC 0095
	JR = K / 100	EQC 0096
	IR = MOD(K, 100)	EQC 0097
	IF (K.EQ.0) GO TO 215	EQC 0098
	IF (JR.EQ.0) GO TO 210	EQC 0099
	$\mathrm{RFRAC}\left(\mathrm{JK},\mathrm{IR}\right)=\mathrm{A/NAEEG}\left(\mathrm{JR},\mathrm{21}\right)$	
	RFRAC(IR, JR) = A/NAREG(IR, 21)	EQC 0101
140	GO TO 215	EQC 0102
	RFRAC(IR, 16) = A/NAREG(IR, 21)	EQC 0103
215	CONTINUE	EQC 0104
	EIGEN=EOLK (21)	EQC 0105
	CALL CYCALC (3, CYBU (21), EIGEN, BP (21), NR, REGBU, REGE, RTYPE,	EQC 0106
	1 NAREG (1,21), RFRAC, BPFRAC (1,21), RLOAD)	EQC 0107
	SUM=0.0	EQC 0108

.

			noc	0 1 0 0
		DO 60 JR = 1, NR		0109
		PSHAR (JR) = REGBU (JR) - BOCBU (JR, 21)		0110
		SUM=SUM+PSHAR(JR) * NAREG(JR, 21)		0111
	60	CONTINUE		0112
		SUM=A NA SS/SUM		0113
		DIF=0.0		0114
		DU 65 JR=1, NR	-	0115
		DIF=AMAX1(DIF,ABS(EOCBU(JR,21)-REGBU(JR)))		0116
	65	PSHAR (JR) = SUM * PSHAR (JR)	-	0117
		IF (DIF.GT.0.01) GO TO 15	EQC	0118
		IF (IS.GT. 10) GO TO 100	EQC	0119
С			EQC	0120
		GO TO (70,80,90), IS	EQC	0121
	70	CONTINUE	EQC	0122
		IF (ITER.GT.1) GO TO 75	EQC	0123
		DIF=100.*(EOLK(21)-EIGEN)	EQC	0124
		OLD=CYBU(21)	EQC	0125
		OLDK=EIGEN	EQC	0126
		CYBU(21) = CYBU(21) - SIGN(AMIN1(2., ABS(DIF)), DIF)	EQC	0127
		GO TO 10	EQC	0128
	75	CONTINUE	EQC	0129
		SLOPE = (CYBU (21) - OLD) / (EIGEN - OLDK)	EQC	0130
		OLDK=EIGEN	EQC	0131
		OLD=CYBU (21)	EQC	0132
		CYBU(21) = CYBU(21) - SLOPE*(EIGEN-EOLK(21))	-	0133
		IF (ABS (CYBU (21) -OLD).LT.0.001) GO TO 150	EQC	0134
		GO TO 10		0135
Ċ			-	0136
-	80	CONTINUE		0137
		IF (ITER.GT.1) GO TO 85		0138
		DIF=1.E+5*(EOLK(21)-EIGEN)		0139
		OLD=BP(21)		0140
		OLDK=EIGEN		0141
		BP (21) = Br (21) - SIGN (AMIN1 (2000., ABS (DIF)), DIF)		0142
		$\frac{1}{3} = \frac{1}{3} = \frac{1}$		0142
	<u>н</u> б	CONTINUE		0145
	05		EQU	0144

		SLOPE = (BP(21) - OLD) / (EIGEN - OLDK)		EQC 0145
		OLDK=EIGEN		EQC 0146
		OLD=BP(21)		EQC 0147
		BP(21) = BP(21) - SLOPE * (EIGEN-EOLK(21))		EQC 0148
		IF (ABS(BP(21)-OLD).LT.1.0) GO TO 150		EQC 0149
		GO TO 10	7	EQC 0150
C				EQC 0151
	90	CONTINUE		EQC 0152
		EOLK(21) = EIGEN		EQC 0153
		GO TO 150		EQC 0154
	100	CONTINUE		EQC 0155
		JS=IS-10		EQC 0156
		IF (1TER.GT. 1) GO TO 105		EQC 0157
		OLD = FRICH (21, JS)		EQC 0158
		OLDK=EIGEN		EQC 0159
		FRICH $(21, JS) = FRICH (21, JS) + 0.05$		EQC 0160
	105	GO TO 10 CONTINUE		EQC 0161
	105			EQC 0162
		SLOPE=(FRICH(21, JS)-OLD)/(EIGEN-OLDK) OLDK=EIGEN		EQC 0163
		OLDEFRICH (21, JS)		EQC 0164
				EQC 0165
		FRICH $(21, JS) = FRICH (21, JS) - SLOPE* (EIGEN-EOLK (21))$ FRICH $(21, JS) = AMN \times 1 (FLIM (1))$ ANTH $1 (FLIM (2))$ FRICH (21, JC))		EQC 0166
		FRICH (21, JS) = AMAX1 (ELIM (1), AMIN1 (ELIM (2), FRICH (21, JS))) IF (ABS (FRICH (21, JS)-OLD).LT.0.001) GO TO 150		EQC 0167
		GO TO 10		EQC 0168
С				EQC 0169
C		CONTINUE		EQC 0170
	400	WRITE (106,101)		EQC 0171
С				EQC 0172
c		END OF ITERATION		EQC 0173
č		DAD OF ITHANIION		EQC 0174
Ū		CONTINJE		EQC 0175
		EOLK(21) = EIGEN		EQC 0176
		CALL CYEDIT (21, REGE, RTYPE)		EQC 0177 EQC 0178
		RETURN		EQC 0178
С				EQC 0179 EQC 0180
-				TAC 0100

.

101 FORMAT(1H0,9X,23HEQ. CYCLE SEARCH FAILED) C

•

.

,

EQC 0181 EQC 0182 EQC 0183

END

SUBROUTINE CYCALC (IS, CYBU, EOLK, BP, NREG, REGBU, REGE, IRTYPE, NAREG,	CYC CYC
1 RFRAC, BPFRAC, RLOAD)	CYC
SUBROUTINE CYCALC - INDIVIDUAL CYCLE CALCULATION WITH OPTIONAL	CYC
SUBROUTINE CICALC - INDIVIDUAL CICLE CALCULATION WITH OFICONAL SEARCH	CYC
SEARCH	CYC
IS SEARCH PARAMETER	CYC
=1 CALCULATE CYCLE BURNUP	CYC
= 2 CLACULATE BP PENALTY	CYC
=2 CLRCULATE EOL K	CYC
	CYC
= 10+ J CALCULATE FEED ENRICHMENT FOR REGION J CYBU CYCLE BURNUP DESIRED OR RETURNED AS CALCULATED	CYC
	~ 110
EOLK EOL K DESIRED OR RETURNED AS CALCULATED REGBU(15) IN AS BOL REGION BURNUPS, OUT AS EOL BURNUPS	CYC
REGE(15) REGION ENRICHMENTS	CYC
	CVC
NAREG (15) NUMBER OF ASSEMBLIES IN EACH REGION RFRAC (15, 16) FRACTION OF EDGES ADJACENT TO BACH REGION ****	CYC
RFRAC (15,16) FRACTION OF EDGES ADJACENT TO BACH REGION ****	СҮС
**** DESTROYED IN ROUTINE *****	CYC
BP NUMBER OF BURNABLE POISON RODS	CYC
NREG NUMBER OF REGIONS	CYC
BPFRAC(15) FRACTION OF BP RODS IN EACH REGION	CYC
RLOAD(15) REGION LOADINGS	CYC
	CYC
COMMON /TABL1/ NBU, NRICH, BURNUP(20), ENRICH(20), ITYPE(20), BIAS(20),	CYC
1 BPWRTH (20), CYBIAS	CYC
COMMON /10/ 105,106,107	CYC
COMMON /TABL2/ KINF (20,20), ISOTOP (20,20,6), POWFIS (20)	CYC
REAL KINF, ISOTOP	СҮС
COMMON /BASIC/ TITLE(20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT,	CYC
1 BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG	CYC
LOGICAL DEBUG	CYC
COMMON /CYCL1/ C1RICH(10),C1TYPE(10),C1ID(10),C1LOAD(10),NAC1(10) COMMON /FEED / NAFEED(21,3),FTYPE(21,3),FLOAD(21,3),FRICH(21,3),	CYC
$\frac{1}{1}$	
1 FID (21, 3)	CYC (

	COMMON /COST / IFCOST	CYC 0037
	COMMON /COEF / COFIT (11), TEMP (2), BWRTH, DPWRTH, XEWRTH	CYC 0038
	COMMON /CYDEP/ BSHARE (15, 10), BBU (10), BKEFF (10), NBBU	CYC 0039
С		CYC 0040
	DIMENSION REGBU(15), REGE(15), IRTYPE(15), NAREG(15), RPRAC(15, 16),	CYC 0041
	1 REGBU1 (15), PSHARE(15)	CYC 0042
	DIMENSION BPFRAC (15), RLOAD (15)	CYC 0043
	REAL*8 A(15,15), B(15), EIGEN	CYC 0044
CDC	REAL A (15, 15), B (15), EIGEN	CYC 0045
	DATA DELBU / 4.0 /	CYC 0046
С		CYC 0047
	IF (IS.EQ.1.AND.CYBU.LE.0.0) CYBU=11.	CYC 0048
	IF (IS.EQ.2.AND.BP.EQ.0.0) $BP=1000$.	CYC 0049
	JS=MOD(IS,10)	CYC 0050
	IF (IS.GT. 10. AND.REGE (JS) $_$ LE.0.0) REGE (JS) = 3.0	CYC 0051
	CALL BUGOUT (8HREGE , REGE, 1, NREG, 15, 1, 1, 1, 2)	CYC 0052
	CALL BUGOUT (8HREGBU, , REGBU, 1, NREG, 15, 1, 1, 1, 2)	CYC 0053
C	CALL BUGOUT (8HRFRAC, , RFRAC, 1, NREG, 15, 1, NREG, 16, 2)	CYC 0054
С	CALL BUGOUT (8HRFRAC , RFRAC, 1, NREG, 15, 16, 16, 16, 2)	CYC 0055
С	CALL BUGOUT (8HNAREG , NAREG, 1, NREG, 15, 1, 1, 1, 1)	CYC 0056
	DO 10 I=1, NR EG	CYC 0057
	B(I) = 0.0	CYC 0058
	DO 5 J=1, NREG	CYC 0059
	RFRAC(I,J) = RFRAC(I,J) * 4.0 * W	CYC 0060
	5 CONTINUE	CYC 0061
	RFRAC(I, I) = RFRAC(I, I) + 1.0 - 4.0 * W * (1.0 + A LPHA * RFRAC(I, 16))	CYC 0062
10	O CONTINUE	CYC 0063
	1 T E R = 0	CYC 0064
C		CYC 0065
1.	5 ITER=ITER+1	CYC 00'66
	IF (ITER.GT.50) GO TO 140	CYC 0067
	DO 20 $I=1$, NREG	CYC 0068
<u>a</u> .	REGBU1(I) = REGBU(I)	CYC 0069
20	0 CONTINUE	CYC 0070
	CYB=0.0	CYC 0071
	IF (IHAL-EQ.1) GO TO 210	CYC 0072

e

		NBBU=0	СҮС	0073
		DO 50 $K=1, 10$	СҮС	0074
		DELB=AMIN1 (DELBU, CYBU-CYB)	СҮС	0075
		PPM=0.0	СҮС	0076
	25	CONTINUE	CYC	0077
		DO 30 I=1, NREG	CYC	0078
		AKINF=TERP (REGBU1(I), REGE(I), IRTYPE(I), KINF, O, AKB, BPK)	CYC	0079
		AKINF=AKINF-BP*BPFRAC(I) *BPK/NAREG(I) +PPM	СҮС	0800
		DO $30 J=1, NREG$	СҮС	0081
		A (I, J) = RFRAC (I, J) * AKINF/AKB	СУС	0082
	-30	CONTINUE	СҮС	0083
С			СҮС	0084
		CALL SOLVE (A, B, EIGEN, NREG)	CYC	0085
С			СҮС	0086
		IF (BWRTH.EQ.0.0) GO TO 34	CYC	0087
		DIF=EIGEN-1.000	СУС	0088
		IF (ABS(DIF).LT.1.E-4) GO TO 34		0089
		PPM=PPM-DIF		0090
		GO TO 25		0091
	34	EIGEN=EIGEN-PPM		0092
		SUM=0.0		0093
		DO 35 I=1, NREG		0094
		B(I)=B(I)*TERP(REGBU1(I),REGE(I),IRTYPE(I),POWFIS,2,AKB,BPK)		0095
		B(I) = B(I) / RLOAD(I)		0096
		SUM = SUM + B(I) + NAREG(I)		0097
	35	CONTINUE		0098
		SUM=SUN/ANASS		0099
		NBBU = NBBU + 1		0100
		DO 40 $I=1$, NR EG		0101
		PSHARE (I) = B(I) / SUM		0102
		BSHARE (I, NBBU) = PSHARE (I)		0103
		REGBU1(I) = REGBU1(I) + DELB + PSHARE(I)		0104
	40	CONTINUE		0105
		BBU(NBBU) = CYB		0106
		BKEFF(NBBU)=EIGEN		0107
		CYB=CYB+DELB	СУС	0108

		IF (CYB.GE.CYBU.AND.DELB.EQ.0.0) GO TO 55	CYC 0109	
		DELB=0.0	CYC 0110	
		IF (CYB.GE.CYBU) GO TO 25	CYC 0111	
	50	CONTINUE	CYC 0112	•
	55	CONTINUE	CYC 0113	6
		GO TO 250	CYC 0114	
	210	CONTINUE	CYC 0115)
С			CYC 0116	
С		HALING DISTRIBUTION CALCULATION	CYC 0117	
С			CYC 0118	r.
		K = 1	CYC 0119	r -
		OLDKH=0.0	CYC 0120	,
		IF (ITER.GT.1) GO TO 220	CYC 0121	
		DO 215 1=1, NREG	CYC 0122	
		PSHARE(I) = 1.0	CYC 0123	
	220	CONTINUE	CYC 0124	
		I TERH=0	CYC 0125	
	225	ITERH=ITERH+1	CYC 0126	
		IF (ITERH.GT.50) GO TO 140	CYC 0127	
		DO 230 I=1, NREG	CYC 0128	
		REGBU1(I) = REGBU (I) + PSHARE(I) * CYBU	CYC 0129	
		AKINF=TERP(REGBU1(I), REGE(I), IRTYPE(I), KINF, O, AKB, BPK)	CYC 0130	
		AKINF=AKINF-BP*BPFRAC(I) *BPK/NAREG(I)	CYC 0131	
		DO 230 J=1, NREG	CYC 0132	
		A (I, J) = RFRAC (I, J) * AKINF/AKB	CYC 0133	
	2.30	CONTINUE	CYC 0134	
С			CYC 0135	
		CALL SOLVE (A, B, EIGEN, NREG)	CYC 0136	
С			CYC 0137	
		SUM=0.0	CYC 0138	
		DO 235 I=1,NREG	CYC 0139	
		B(I) = B(I) * TERP(REGBU1(I), REGE(I), IRTYPE(I), POWFIS, 2, AKB, BPK)	CYC 0140	
		B(I) = B(I) / R LOAD(I)	CYC 0141	
		SUM=SUM+B(I)*NAREG(I)	CYC 0142	
	235	CONTINUE	CYC 0143	
		SUM=SUM/ANASS	CYC 0144	

		DIFH=0.0	CYC	0145
		DO 240 I=1,NREG	CYC	0146
		PSHARE(I) = B(I) / SUM	CYC	0147
		BSHARE (I,K)=PSHARE (I)	СҮС	0 148
		DIFH=AMAX1 (DIFH, ABS (REGBU (I) +PSHARE (I) *CYBU-REGBU1 (I)))	СУС	0149
	240	CONTINUE	CYC	0150
		NBBU=1	CYC	0151
		BBU(1) = CYBU	СҮС	0152
		BKEFF(1) = EIGEN	CYC	0153
		IF (DIFH.GT.0.001) GO TO 225	CYC	0154
	250	CONTINUE	СУС	0155
		1F (IS.GT.10) GO TO 90	СҮС	0156
		GO TO (60,70,80), IS	СҮС	0157
	60	CONTINUE	CYC	0158
С			CYC	0159
С		SEARCH ON CYCLE BURNUP	CYC	0160
С			CYC	0161
		IF (ITER.GT.1) GO TO 65	CYC	0162
		DIF=100.0*(EOLK-EIGEN)	СУС	0163
		O L D = CY BU	СҮС	0164
		OLDK=EIGEN	CYC	0165
		CY BU=CYBU - SIGN (AMIN1(2. ,ABS (DIF)),DIF)	CYC	0166
		GO TO 15	СҮС	0167
	65	CONTINUE	CYC	0168
		SLOPE= (CYBU-OLD) / (EIGEN-OLDK)	СҮС	0169
		OLDK=LIGEN	CYC	0170
		O L D = C Y B U		0171
		CYBU=CYBU-SLOPE* (EIGEN-EOLK)		0172
		IF (ABS(CYBU-OLD).LT.0.001) GO TO 150		0173
		GO TO 15		0174
	70	CONTINUE		0175
С				0176
С		SEARCH ON BP PENALTY		0177
С				0178
		LF (ITER.GT.1) GO TO 75		0179
		DIF=1.E+5* (EOLK-EIGEN)		0 180
		- · ·		

		•	
		OLD=BP	CYC 0181
		OLDK= EIGEN	CYC 0182
		BP=BP-SIGN (AMIN1(2000.,ABS(DIF)),DIF)	CYC 0183
	7.0	GO TO 15	CYC 0184
	15		CYC 0185
		SLOPE = (BP-OLD) / (ElGEN-OLDK)	CYC 0186
		OLDK=EIGEN	CYC 0187
		O LD = B P	CYC 0188
		BP=BP-SLOPE*(EIGEN-EOLK)	CYC 0189
		IF (ABS (BP-OLD).LT. 1.0) GO TO 150	CYC 0190
	() ()	GO TO 15	CYC 0191
~	80	CONTINUE	СУС 0192
С			CYC 0193
C		CALCULATE EOLK	CYC 0194
С			CYC 0195
		EOLK=EIGEN	CYC 0196
	60	GO TO 150	CYC 0197
	90	CONTINUE	CYC 0198
C		203 00/ 01 01 01 01 00 00	CYC 0199
Ć		SEARCH ON ENTICHMENT	CYC 0200
С		TR (TRUER OF 1) OF 65	CYC 0201
		IF (ITER.GT.1) GO TO 95	CYC 0202
		OLD=REGE (JS)	СУС 0203
		OLDK=EIGEN	CYC 0204
		REGE (JS) = REGE (JS) + 0.10	СУС 0205
	0 5	GO TO 15	СУС 0206
	95	CONTINUE	СУС 0207
		SLOP L = (REGE (JS) - OLD) / (EIGEN - OLDK)	CYC 0208
		OLDK=EIGEN	CYC 0209
		OLD=REGE (JS)	CYC 0210
		REGE (JS) =REGE (JS) -SLOPE* (EIGEN-EOLK)	CYC 0211
		REGE(JS) = AMAX1(ELIM(1), AMIN1(ELIM(2), REGE(JS)))	CYC 0212
		IF (ABS (REGE (JS) - OLD) . LT. 0.001) GO TO 150	CYC 0213
~		GO TO 15	CYC 0214
C			CYC 0215
Û		END OF ITERATION LOOP	CYC 0216

С		CYC 0217
	140 CONTINUE	CYC 0218
	WRITE (IO6,101) IS, CYBU, EOLK	CYC 0219
	150 CONTINUE	CYC 0220
	EOLK=EIGEN	CYC 0221
	DO 155 1=1,NREG	CYC 0222
	REGBU(I) = REGBU1(I)	CYC 0223
	155 CONTINUE	CYC 0224
	RETURN	CYC 0225
С		CYC 0226
	101 FORMAT (1H0, 10X, 31HCYCLE ITERATION PAILED, SEARCH=, 12, 4H BU=, F9.0	CYC 0227
	1 $6H EOLK=, F9.5 /)$	CYC 0228
С		CYC 0229
	END	CYC 0230

SUBROUTINE CYLDIT (IC, REGE, IR TYPE)	CYED0001
	CYED0002
EDITS OUT CYCLE INFORMATION	CYED0003
	CYED0004
COMMON /TABL1/ NBU, NRICH, BURNUP(20), ENRICH(20), ITYPE(20), BIAS(20),	CYED0005
1 BPWRTH (20), CY BIAS	CYED0006
COMMON /10/ 105,106,107	CYED0007
COMMON /TABL2/ KINF(20,20), I SOTOP(20,20,6), POWFIS(20)	CYED0008
RÉAL KINF, ISOTOP	CYED0009
COMMON /BASIC/ TITLE(20),NASS,ANASS,NADIAM,NALPHA,ULOAD,HEIGHT,	CYED0010
1 BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG	CYED0011
LOGICAL DEBUG	CY ED0012
COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21),	CYED0013
1 NFEED(21), NREG(21), CYLOAD(21), CAPFAC(21), CYTIM(21),	CYED0014
2 SDLEN (21), IFEDGE (21)	CYED0015
COMMON /RGDAT/ NEDGE(2,1000), IEDGE(21), NAREG(15,21), JSHUF(15,21),	CYED0016
1 BPFRAC(15,21), BOCBU(15,21), EOCBU(15,21),	CYED0017
2 IDXREG(15,21), NADIS(15,21)	CYED0018
COMMON /CYCL1/ C1RICH(10),C1TYPE(10),C1ID(10),C1LOAD(10),NAC1(10)	CYED0019
COMMON /FEED / NAFEED (21,3), FTYPE (21,3), FLOAD (21,3), FRICH (21,3),	CYED0020
1 FID(21,3)	CY ED00 2 1
INTEGER FTYPE,C1TYPE	CYED0022
COMMON /COST / IFCOST	CYED0023
COMMON /COEF / COFIT(11),TEMP(2),BWRTH,DPWRTH,XEWRTH	CYED0024
COMMON /EDGE/ NAE(15,16)	CYED0025
COMMON /CYDEP/ BSHARE(15,10),BBU(10),BKEFF(10),NBBU	CYED0026
	CYED0027
DIMENSION REGE(15), IRTYPE(15), KSUR(2,6), RD(15)	CYED0028
	CYED0029
DATA KSUR / 4HBURN, 4HUP , 4HNO. , 3HBPS, 4HEOL , 4HKEFF, 4HFEED,	CYED0030
1 2H 1, 4HFEED, 2H 2, 4HFEED, 2H 3 /	CYED0031
	CYED0032
CALL HEAD	CYED0033
WRITE (106,101) IC	CYED0034
IS=ISURCH (IC)	CYED0035
IF (IS.GT.10) IS=IS-7	CYED0036

С

С

С

1 CYBU(IC)*1.E+5*CYLOAD(IC)/(POWER*CYTIM(IC)) IF (CAPFAC(IC)=20.0.0) CAPFAC(IC)=80. CYED00 CTT1M(IC)=CYBU(IC)*CYLOAD(IC) *1.E*5/(POWER*CAPFAC(IC)) EFPH=0.01*24.*CAPFAC(IC)*CYTIM(IC) WRITE (IC6,103) CYBU(IC).EFPH,CYTIM(IC),SDLEN(IC),CAPFAC(IC), CYED00 NR=WREG(IC) WRITE (IC6,104)(I,I=1,NR) WRITE (IC6,105) (REGE(I),I=1,NR) WRITE (IC6,106) (IRTYPE(1),I=1,NR) WRITE (IC6,106) (IRTYPE(1),I=1,NR) WRITE (IC6,106) (IRTYPE(1),I=1,NR) WRITE (IC6,106) (IRTYPE(1),I=1,NR) WRITE (IC6,106) (IRTYPE(1),I=1,NR) WRITE (IC6,107) (BPFAC(I,IC),I=1,NR) WRITE (IC6,109) (NADIS(I,IC),I=1,NR) WRITE (IC6,109) (NADIS(I,IC),I=1,NR) CYED00 WRITE (IC6,109) (NADIS(I,IC),I=1,NR) CYED00 J1=IJNRB CYED00 J1=IJNRB CYED00 J1=IJNRB CYED00 J1=IJNRB CYED00 J1=IJNRB CYED00 J1=C(I,IC) J2=MOD(II,100) J2=MOD(II,100) S KD(I)=CILD(J2) CYED00 WRITE (IC6,111) (RD(I),I=1,NR) CYED00 WRITE (IC6,111) (RD(I),I=1,NR) CYED00 CYED00 WRITE (IC6,115) BBU(J),BKEFF(J),(BSHARE(I,J),I=1,NR) CYED00		KSUR(I,IS),I=1,2)	CYED0037
<pre>LF (CAPFAC(IC) = EQ. 0. 0) CAPFAC(IC) = 80. CYT1M(IC) = CYBU(IC) * CYLOAD(IC) *1. B*5/(POWER* CAPFAC(IC)) CYEDDO EPPH=0.01*24.*CAPFAC(IC) * CYLOAD(IC) CYEDDO wRITE (IO6, 103) CYBU(IC), EPPH, CYTIM(IC), SDLEN(IC), CAPFAC(IC), CYEDDO N= #AEEG(IC) CYEDOD WRITE (IO6, 104) (1, I=1, NR) CYEDOD WRITE (IO6, 105) (REGE(I), I=1, NR) CYEDOD WRITE (IO6, 106) (IRTYPE(1), I=1, NR) CYEDOD WRITE (IO6, 106) (IRTYPE(1), I=1, NR) CYEDOD WRITE (IO6, 108) (NAREG(I,IC), I=1, NR) CYEDOD WRITE (IO6, 109) (NAREG(I,IC), I=1, NR) CYEDOD WRITE (IO6, 109) (NAREG(I,IC), I=1, NR) CYEDOD WRITE (IO6, 109) (NAREG(I,IC), I=1, NR) CYEDOD WRITE (IO6, 100) (INTY) (IST) (IS</pre>			CYED0038
CYTIA (IC) = CYBÜ (IČ) *CYLOAD (IC) *1.±+5/(POWER*CAPPAC (IC)) EPPH=0.01*24.*CAPPAC (IC) *CYTIM (IC) WRITE (IO6,103) CYBU (IC), EPPH.CYTIM (IC), SDLEN (IC), CAPPAC (IC), EOLK (IC), bP (IC), CYLOAD (IC) NH= AREG (IC) WRITE (IO6,104) (I, I=1,NR) WRITE (IO6,105) (REGE (I), I=1,NR) WRITE (IO6,105) (REGE (I), I=1,NR) CYEDOO WRITE (IO6,106) (IRTYPE(I),I=1,NR) WRITE (IO6,109) (NAREG (I,IC),I=1,NR) WRITE (IO6,109) (NAREG (I,IC),I=1,NR) WRITE (IO6,109) (NAREG (I,IC),I=1,NR) WRITE (IO6,109) (NAREG (I,IC),I=1,NR) WRITE (IO6,109) (NADIS (I,IC),I=1,NR) WRITE (IO6,109) (NADIS (I,IC),I=1,NR) WRITE (IO6,109) (NADIS (I,IC),I=1,NR) CYEDOO WRITE (IO6,109) (NADIS (I,IC),I=1,NR) CYEDOO WRITE (IO6,109) (NADIS (I,IC),I=1,NR) CYEDOO WRITE (IO6,109) (IL) WRITE (IO6,109) (IL) WRITE (IO6,109) (IL) WRITE (IO6,109) (IL) WRITE (IO6,109) (IL) CYEDOO CYEDOO J2=NOD (IL,100) CYEDOO S KD (I)=CILD (J2) CONTINUE WRITE (IO6,111) (RD (I),I=1,NR) WRITE (IO6,111) (RD (I),I=1,NR) WRITE (IO6,111) (RD (I),I=1,NR) WRITE (IO6,111) (RD (I),I=1,NR) WRITE (IO6,114) (I,IC),I=1,NR) WRITE (IO6,115) BBU (J),BREFF (J),(BSHARE (I,J),I=1,NR) CYEDOO 20 CONTINUE CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO WRITE (IO6,115) BBU (J),BREFF (J),(BSHARE (I,J),I=1,NR) CYEDOO C			CYED0039
EFPH=0.01*24.*CAPPAC(IC)*CYTIM(IC) CYED00 wRITE (I06,103) CYBU(IC), EPPH,CTIIM(IC), SDLEN(IC), CAPPAC(IC), CYED00 1 EOLK(IC), BP(IC), CYLOAD(IC) CYED00 NR=NREG(IC) CYED00 wRITE (I06,104) (I,I=1,NR) CYED00 wRITE (I06,105) (REGE(I),I=1,NR) CYED00 wRITE (I06,106) (IRTYPE(I),I=1,NR) CYED00 wRITE (I06,108) (NAREG(I,IC),I=1,NR) CYED00 wRITE (I06,109) (NALEG(I,IC),I=1,NR) CYED00 wRITE (I06,109) (NALEG(I,IC),I=1,NR) CYED00 wRITE (I06,110) (JSHUP(I,IC),I=1,NR) CYED00 wkITE (I06,110) (JSHUP(I,IC),I=1,NR) CYED00 Du 10 I=1,NR CYED00 J1=IL/100 CYED00 J2=NOD (II,100) CYED00 LF (I0,1,11) (GD TO 5 CYED00 Rb (I)=C11D (J2) CYED00 UG CONTINUE CYED00 WRITE (I06,111) (RD (I),I=1,NR) CYED00 WRITE (I06,111) (BCBU(I,IC),I=1,NR) <td></td> <td>• • •</td> <td>CYED0040</td>		• • •	CYED0040
wRITE (I06,103) CYB0(IC), EPPH, CYTIM (IC), SDLEN (IC), CAPFAC (IC), CYED00 1 EOLK (IC), EP (IC), CYLOAD (IC) CYED00 NH=NREG (IC) CYED00 WRITE (I06,104) (I, I=1,NR) CYED00 WRITE (I06,105) (REGE (I), I=1,NR) CYED00 WRITE (I06,106) (IRTYPE (I), I=1,NR) CYED00 WRITE (I06,106) (IRTYPE (I), I=1,NR) CYED00 WRITE (I06,109) (NAEG (I,IC), I=1,NR) CYED00 WRITE (I06,109) (NAEG (I,IC), I=1,NR) CYED00 WRITE (I06,109) (NADIS (I,C), I=1,NR) CYED00 WRITE (I06,100) (JSHUP(I,IC), I=1,NR) CYED00 DO 10 I=1,NR CYED00 II=IDXREG (I,IC) CYED00 J2=H0D (II,100) CYED00 J2=H0D (II,100) CYED00 CYED00 CYED00 J2=HD0 (I1,100) CYED00 CYED00 CYED00 J2=H0D (I1,100) CYED00 CYED00 CYED00 J0 CONTINUE CYED00 WRITE (I06,111) (RD (I),I=1,NR) CYED00 WRITE (I06,111) (RD (I),I=1,NR) CYED00 WRITE (I06,113) (EOCBU (I,IC),I=1,NR) CYED00 WRITE (I06,114) (I,I=1			CY ED0041
1 EOLK (IC), BP (IC), CYLOAD (IC) CYED00 NH= NREG (IC) CYED00 WITE CYED00 WITE (IO6,104) (I, I=1,NR) CYED00 WRITE (IO6,105) (REGE (I), I=1,NR) CYED00 WRITE (IO6,105) (REYPE(I), I=1,NR) CYED00 WRITE (IO6,106) (IRYYPE(I), I=1,NR) CYED00 WRITE (IO6,108) (NAREG (I, IC), I=1,NR) CYED00 WRITE (IO6,109) (NADIS (I, IC), I=1,NR) CYED00 WRITE (IO6,109) (NADIS (I, IC), I=1,NR) CYED00 WRITE (IO6,109) (NADIS (I, IC), I=1,NR) CYED00 Do 1 o I=1,NR CYED00 II=IDXREG (I, IC) I=1,NR) CYED00 J2=MOD (II, 100) ISHUF (I, IC), I=1,NR) CYED00 J2=MOD (II, 100) CYED00 CYED00 J2=MOD (II, 100) CYED00 CYED00 J2=MOD (II, 100) CYED00 CYED00 J0 CONTINUE CYED00 CYED00 WRITE (IO6,111) (RD (I), I=1,NR) CYED00 WRITE (IO6,112) (BOCBU (I, IC), I=1,NR) CYED00 WRITE (IO6,113) (EOCBU (I, IC), I=1,NR) CYED00 WRITE (IO6,114) (I, I=1,NR)			CYED0042
NR=NREG(IC) CYED00 WRITE (I06,104) (I, I=1,NR) CYED00 WRITE (I06,105) (REGE(I),I=1,NR) CYED00 WRITE (I06,106) (RTYPE(I),I=1,NR) CYED00 IF (BP(IC).GT.U.O) WRITE (I06,107) (BPFRAC(I,IC),I=1,NR) CYED00 WRITE (I06,108) (NARLE(I,IC),I=1,NR) CYED00 WRITE (I06,109) (NARLE(I,IC),I=1,NR) CYED00 WRITE (I06,110) (JSHUF(I,IC),I=1,NR) CYED00 Du 10 I=1,NR CYED00 J1=IDXREG(I,IC) CYED00 J2=HOD(II,100) CYED00 J2=HOD(II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 RU (I)=FID (J1,J2) CYED00 WRITE (I06,111) (RD(I),I=1,NR) CYED00 WRITE (I06,111) (RD(I),I=1,NR) CYED00 WRITE (I06,111) (RD(I),I=1,NR) CYED00 WRITE (I06,111) (RD(I),I=1,NR) CYED00 WRITE (I06,113) (FOCBU(I,IC),I=1,NR) CYED00 WRITE (I06,114) (RD(I),I=1,NR) CYED00 WRITE (I06,113) (FOCBU(I,IC),I=1,NR) CYED00 WRITE (I06,113) (FOCBU(I,IC),I=1,NR) CYED00 D0 20 J=1,NBBU CYED00 WRITE (I06,114) (I,I=1,NR) CYED00	-		CYED0043
WRITE (106,104) (1, I=1,NR) CYED00 WRITE (106,105) (REGE (1), I=1,NR) CYED00 WRITE (106,106) (IRTYPE(1), I=1,NR) CYED00 IF (BP (1C).GT.U.0) WRITE (106,107) (BPPRAC (I,IC), I=1,NR) CYED00 WRITE (106,108) (NAREG (I,IC), I=1,NR) CYED00 WRITE (106,109) (NADIS (I,IC), I=1,NR) CYED00 WRITE (106,110) (JSHUF(I,IC), I=1,NR) CYED00 DO 10 I=1,NR CYED00 I1=IDXREG (I,IC) II=1,NR CYED00 J1=II/100 J2=NOD (II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 RD (1)=CIID (J1,32) CYED00 GO TO 10 CYED00 Sk D(1)=CIID (J2) CYED00 WRITE (106,111) (RD (I), I=1,NR) CYED00 WRITE (106,111) (RD (I), I=1,NR) CYED00 WRITE (106,112) (BOCBU(I,IC), I=1,NR) CYED00 WRITE (106,113) (EOCBU(I,IC), I=1,NR) CYED00 WRITE (106,115) BBU (J), BKEFF (J		IC) , BP (IC) , CYLOAD (IC)	CYED0044
WRITE (I06,105) (REGE (I), I=1,NR) CYED00 WRITE (I06,106) (IRTYPE(I),I=1,NR) CYED00 IF (BP(IC).GT.U.O) WRITE (I06,107) (BPFRAC(I,IC),I=1,NR) CYED00 WRITE (I06,109) (NALGE (I,IC),I=1,NR) CYED00 WRITE (I06,109) (NADIS (I,IC),I=1,NR) CYED00 WRITE (I06,109) (NADIS (I,IC),I=1,NR) CYED00 WRITE (I06,110) (JSHUF(I,IC),I=1,NR) CYED00 DO 10 I=1,NR CYED00 II=IDXREG (I,IC) CYED00 J=II/100 CYED00 J2=MOD (II,100) CYED00 CYED00 CYED00 J1=IL/100 CYED00 J2=MOD (II,100) CYED00 CYED00 CYED00 J0 (J)=FID (J1,J2) CYED00 G0 T0 10 CYED00 SkD (I)=C11D (J2) CYED00 VRITE (I06,111) (RD (I),I=1,NR) CYED00 WRITE (I06,112) (BOCBU (I,IC),I=1,NR) CYED00 WRITE (I06,113) (EOCBU (I,IC),I=1,NR) CYED00 WRITE (I06,113) (EOCBU (I,IC),I=1,NR) CYED00 D0 20 J=1,NBBU CYED00 WRITE (I06,114) (I,I=1,NR) CYED00 D0 20 J=1,NBBU CYED00			CYED0045
WRITE (I06,106) (IRTYPE(I), I=1,NR) CYED00 IF (BP(IC).GT.0.0) WRITE (I06,107) (BPFRAC(I,IC),I=1,NR) CYED00 WRITE (I06,108) (NAREG(I,IC),I=1,NR) CYED00 WRITE (I06,109) (NADIS(I,IC),I=1,NR) CYED00 WRITE (I06,110) (JSHUF(I,IC),I=1,NR) CYED00 D0 10 I=1,NR CYED00 J1=IL/100 CYED00 J2=M0D(II,100) CYED00 CYED00 CYED00 SRD(I)=FID(J1,J2) CYED00 CYED00 CYED00 SRD(I)=C1ID(J2) CYED00 WRITE (I06,111) (RD(I), I=1,NR) CYED00 WRITE (I06,113) (EOCBU(I,IC), I=1,NR) CYED00 WRITE (I06,113) (EOCBU(I,IC), I=1,NR) CYED00 WRITE (I06,114) (I,I=1,NR) CYED00 WRITE (I06,115) BBU (J), BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 20 CONTINUE CYED00			CYED0046
IF (BP (IC).GT.0.0) WRITE (I06,107) (BPFRAC (I,IC),I=1,NR) CYED00 WRITE (I06,108) (NAREG (I,IC),I=1,NR) CYED00 WRITE (I06,109) (NADIS (I,IC),I=1,NR) CYED00 Do 10 I=1,NR CYED00 J1=IDXREG (I,IC) J1=1,NR CYED00 J2=MOD (II,100) CYED00 CYED00 J1=II/100 CYED00 CYED00 J2=MOD (II,100) CYED00 CYED00 J1=II/100 CYED00 CYED00 J2=MOD (II,100) CYED00 CYED00 J2=MOD (II,100) CYED00 CYED00 J0 GO TO 5 CYED00 RD (I)=FID (J1,J2) CYED00 CYED00 GO TO 10 CYED00 CYED00 S RD (I)=C11D (J2) CYED00 CYED00 WRITE (I06,111) (RD (I),I=1,NR) CYED00 CYED00 WRITE (I06,112) (BOCBU (I,IC),I=1,NR) CYED00 CYED00 WRITE (I06,113) (EOCBU (I,IC),I=1,NR) CYED00 CYED00 WRITE (I06,114) (I,I=1,NR) CYED00 CYED00 WRITE (I06,115) BBU (J),BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 WRITE (I06,115) BBU (J),BKEFF (J), (BSHARE (I,J),I=1,NR)			CYED0047
WRITE (I06,108) (NAREG (I,IC),I=1,NR) CYED00 WRITE (I06,109) (NADIS (I,IC),I=1,NR) CYED00 WRITE (I06,110) (JSHUF(I,IC),I=1,NR) CYED00 Do 10 I=1,NR CYED00 II=IDXREG (I,IC) CYED00 J1=II/100 CYED00 J2=MOD (II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 RD (I)=FID (J1,J2) CYED00 GO TO 10 CYED00 SkD (I)=C11D (J2) CYED00 WRITE (I06,111) (RD (I),I=1,NR) CYED00 WRITE (I06,111) (RD (I),I=1,NR) CYED00 WRITE (I06,112) (BOCBU (I,IC),I=1,NR) CYED00 WRITE (I06,113) (FOCBU (I,IC),I=1,NR) CYED00 WRITE (I06,114) (I,I=1,NR) CYED00 WRITE (I06,115) BBU (J),BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 20 CONTINUE CYED00			CYED0048
WRITE (I06,109) (NADIS (I,IC),I=1,NR) CYED00 WKITE (I06,110) (JSHUF(I,IC),I=1,NR) CYED00 D0 10 I=1,NR CYED00 I1=IDXREG (I,IC) CYED00 J=II/100 CYED00 J2=MOD (II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 GO TO 10 CYED00 S RD (I)=FID (J1,J2) CYED00 GO TO 10 CYED00 S RD (I)=C1ID (J2) CYED00 WRITE (I06,111) (RD (I),I=1,NR) CYED00 WRITE (I06,112) (BOCBU (I,IC),I=1,NR) CYED00 WRITE (I06,113) (EOCBU (I,IC),I=1,NR) CYED00 WRITE (I06,114) (I,I=1,NR) CYED00 WRITE (I06,115) BBU (J),BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 20 CONTINUE CYED00			CYED0049
WRITE (I06,110) (JSHUF(I,IC),I=1,NR) CYED00 D0 10 I=1,NR CYED00 II=DXREG(I,IC) CYED00 J1=II/100 CYED00 J2=MOD(II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 GO TO 10 CYED00 Sk D(I)=C11D(J2) CYED00 WRITE (I06,111) (RD(I),I=1,NR) CYED00 WRITE (I06,112) (BOCBU(I,IC),I=1,NR) CYED00 WRITE (I06,113) (EOCBU(I,IC),I=1,NR) CYED00 WRITE (I06,114) (I,I=1,NR) CYED00 WRITE (I06,115) BBU(J),BKEFF(J),(BSHARE(I,J),I=1,NR) CYED00 WRITE (106,115) BBU(J),BKEFF(J),(BSHARE(I,J),I=1,NR) CYED00 20 CONTINUE CYED00			CYED0050
Do 10 I=1,NR CYED00 II=IDXREG (I,IC) CYED00 J1=II/100 CYED00 J2=MOD (II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 GO TO 10 CYED00 CYED00 S RD (I)=C1ID (J2) CYED00 CYED00 10 CONTINUE CYED00 WRITE (IO6,111) (RD (I), I=1,NR) CYED00 WRITE (IO6,112) (BOCBU (I,IC), I=1,NR) CYED00 WRITE (IO6,113) (EOCBU (I,IC), I=1,NR) CYED00 WRITE (IO6,114) (I,I=1,NR) CYED00 D0 20 J=1,NBBU CYED00 WRITE (106,115) BBU (J), BKEFF (J), (BSHARE (I,J), I=1,NR) CYED00 20 CONTINUE CYED00			CYED0051
II=IDXREG(I,IC) CYED00 J1=II/100 CYED00 J2=MOD(II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 GO TO 10 CYED00 S KD(I)=C11D(J2) CYED00 10 CONTINUE CYED00 WRITE (IO6,111) (RD(I),I=1,NR) CYED00 WRITE (IO6,112) (BOCBU(I,IC),I=1,NR) CYED00 WRITE (IO6,113) (EOCBU(I,IC),I=1,NR) CYED00 WRITE (IO6,114) (I,I=1,NR) CYED00 WRITE (IO6,115) BBU (J),BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 20 CONTINUE CYED00		JSHUF(I,IC), I=1, NR)	CYED0052
J1=II/100 CYED00 J2=MOD (II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 RD (I)=FID (J1,J2) CYED00 GO TO 10 CYED00 10 CONTINUE CYED00 WRITE (IO6,111) (RD (I),I=1,NR) CYED00 WRITE (IO6,112) (BOCBU (I,IC),I=1,NR) CYED00 WRITE (IO6,113) (EOCBU (I,IC),I=1,NR) CYED00 WRITE (IO6,114) (I,I=1,NR) CYED00 WRITE (IO6,115) BBU (J),BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 20 CONTINUE CYED00			
J2=MOD (II,100) CYED00 IF (J1.EQ.1) GO TO 5 CYED00 RD (I)=FID (J1,J2) CYED00 GO TO 10 CYED00 5 RD (I)=C1ID (J2) CYED00 10 CONTINUE CYED00 WRITE (IO6,111) (RD (I),I=1,NR) CYED00 WRITE (IO6,112) (BOCBU (I,IC),I=1,NR) CYED00 WRITE (IO6,113) (EOCBU (I,IC),I=1,NR) CYED00 WRITE (IO6,114) (I,I=1,NR) CYED00 WRITE (IO6,114) (I,I=1,NR) CYED00 WRITE (IO6,115) BBU (J),BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 20 CONTINUE CYED00	- · · ·		CYED0054
IF (J1.EQ.1) GO TO 5 CYEDOO RD (I) = FID (J1,J2) CYEDOO GO TO 10 CYEDOO CYEDOO 5 RD (I) = C1ID (J2) 10 CONTINUE CYEDOO WRITE (IO6,111) (RD (I), I=1,NR) CYEDOO WRITE (IO6,112) (BOCBU (I,IC), I=1,NR) CYEDOO WRITE (IO6,113) (EOCBU (I,IC), I=1,NR) CYEDOO WRITE (IO6,114) (I,I=1,NR) CYEDOO DO 20 J=1,NBBU CYEDOO WRITE (IO6,115) BBU (J), BKEFF (J), (BSHARE (I,J), I=1,NR) 20 CONTINUE CYEDOO CYEDOO CYEDOO	•		CYED0055
RD (I) = FID (J1, J2) CYED00 GO TO 10 CYED00 5 RD (I) = C1ID (J2) CYED00 10 CONTINUE CYED00 WRITE (IO6, 111) (RD (I), I=1, NR) CYED00 WRITE (IO6, 112) (BOCBU (I, IC), I=1, NR) CYED00 WRITE (IO6, 113) (EOCBU (I, IC), I=1, NR) CYED00 WRITE (IO6, 114) (I, I=1, NR) CYED00 WRITE (IO6, 114) (I, I=1, NR) CYED00 DO 20 J=1, NBBU CYED00 WRITE (IO6, 115) BBU (J), BKEFF (J), (BSHARE (I, J), I=1, NR) CYED00 20 CONTINUE CYED00	• • •		CYED0056
GO TO 10 CYED00 5 RD(I)=C1ID(J2) CYED00 10 CONTINUE CYED00 WRITE (IO6,111) (RD(I),I=1,NR) CYED00 WRITE (IO6,112) (BOCBU(I,IC),I=1,NR) CYED00 WRITE (IO6,113) (EOCBU(I,IC),I=1,NR) CYED00 WRITE (IO6,114) (I,I=1,NR) CYED00 WRITE (IO6,115) BBU(J),BKEFF(J), (BSHARE(I,J),I=1,NR) CYED00 20 CONTINUE CYED00		0 5	CYED0057
5 RD (I) = C 1 ID (J2) CYED00 10 CONTINUE CYED00 WRITE (IO6,111) (RD (I),I=1,NR) CYED00 WRITE (IO6,112) (BOCBU (I,IC),I=1,NR) CYED00 WRITE (IO6,113) (EOCBU (I,IC),I=1,NR) CYED00 WRITE (IO6,114) (I,I=1,NR) CYED00 WRITE (IO6,115) BBU (J), BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 20 CONTINUE CYED00			CYED0058
10 CONTINUE CYEDO0 WRITE (I06,111) (RD(I),I=1,NR) CYEDO0 WRITE (I06,112) (BOCBU(I,IC),I=1,NR) CYEDO0 WRITE (I06,113) (EOCBU(I,IC),I=1,NR) CYEDO0 WRITE (I06,114) (I,I=1,NR) CYEDO0 DO 20 J=1,NBBU CYEDO0 WRITE (I06,115) BBU(J),BKEFF(J), (BSHARE(I,J),I=1,NR) CYEDO0 20 CONTINUE CYEDO0			CYED0059
WRITE (I06,111) (RD(I),I=1,NR) CYEDOO WRITE (I06,112) (BOCBU(I,IC),I=1,NR) CYEDOO WRITE (I06,113) (EOCBU(I,IC),I=1,NR) CYEDOO WRITE (I06,114) (I,I=1,NR) CYEDOO DO 20 J=1,NBBU CYEDOO WRITE (I06,115) BBU(J),BKEFF(J), (BSHARE(I,J),I=1,NR) CYEDOO CYEDOO CYEDOO CYEDOO CYEDOO WRITE (I06,115) BBU(J), BKEFF(J), (BSHARE(I,J),I=1,NR) CYEDOO CYEDOO CYEDOO			CYED0060
WRITE (106,112) (BOCBU(I,IC),I=1,NR) CYEDOO WRITE (106,113) (EOCBU(I,IC),I=1,NR) CYEDOO WRITE (106,114) (I,I=1,NR) CYEDOO DO 20 J=1,NBBU CYEDOO WRITE (106,115) BBU(J),BKEFF(J), (BSHARE(I,J),I=1,NR) CYEDOO 20 CONTINUE CYEDOO			CYED0061
WRITE (I06, 113) (EOCBU(I,IC),I=1,NR) CYEDOO WRITE (I06, 114) (I,I=1,NR) CYEDOO DO 20 J=1,NBBU CYEDOO WRITE (106, 115) BBU(J),BKEFF(J), (BSHARE(I,J),I=1,NR) CYEDOO 20 CONTINUE CYEDOO			CYED0062
wRITE (I06,114) (I,I=1,NR) CYED00 D0 20 J=1,NBBU CYED00 WRITE (106,115) BBU (J),BKEFF (J), (BSHARE (I,J),I=1,NR) CYED00 20 CONTINUE CYED00			CYED0063
DO 20 J=1, NBBU CYED00 WRITE (106,115) BBU (J), BKEFF (J), (BSHARE (I,J), I=1, NR) CYED00 20 CONTINUE CYED00			CYEDÜÖ64
WRITE (106,115) BBU (J), BKEFF (J), (BSHARE (I,J), I=1, NR) CYEDOO 20 CONTINUE CYEDOO		I, I=1, NR)	CYED0065
20 CONTINUE CYEDOO	•		CYED0066
		BU (J) , BKEFF (J) , (BSHARE (I,J) , $I=1$, NR)	CY ED0067
DO 25 I=1.16 CYEDOO			CYED0068
	DO 25 I=1,16		CYED0069
			CYED0070
	· · · /		CYED0071
11=0 . CYED00	I 1 = 0		CYED0072

	IF (IC.GT.1) I1=IEDGE(IC-1)	CYED0073
	11=11+1	CYED0074
	12=1EDGE(IC)	CYED0075
	DO 35 I=11, I2	CYED0076
	JK = NEDGE(1, I)	CYED0077
	J = JK / 100	CYED0078
	K=MOD (JK, 100)	CYED0079
	IF (JK.EQ.0) GO TO 35	CYEDŬ080
	1F (J.EQ.0) GO TO 30	CY ED0081
	NAE (J,K) = NEDGE (2,1)	CYED0082
	NAE (S, K) = NEDGE (2, 1) $NAE (K, J) = NEDGE (2, 1)$	CYED0083
	GO = TO = 35	CYED0084
3.0	(K, 16) = NEDGE(2, I)	CYED0085
	CONTINUE	CYED0086
J.	WRITE $(106, 116)$ $(1, 1=1, NR)$	CYED0087
	DO 50 $JR=1$, NR	CYED0088
	N=0	CYED0089
	DU 40 I = 1, 16	CYED0090
<u>ц</u> (N=N+NAE(JR,I)	CYED0091
	$N = 4 \times NAREG (JR, IC)$	CYED0092
	WRITE (106, 117) JR, NN, N, NAE(JR, 16), (NAE(JR, I), 1=1, NR)	CYED0093
5(CONTINUE	CYED0094
	RETURN	CYED0095
С		CYED0096
	FORMAT (30X, 5HCYCLE, I3)	CYED0097
	P FORMAT (1H0, 29X, 18HSEARCH PARAMETER, 2A4)	CYED0098
	FORNAT (1H0, 29X, 18HBURNUP	CYED0099
	1 30X, 18HEFPH	C YEDO 100
	2 30X, 18HCYCLE TIME	CYED0101
	3 30X, 18HS HUT DOWN LENGTH, F7. 1, 5H DAYS /	CYED0102
	3 30X, 18HCAPACITY FACTOR, F7. 1, 8H PERCENT /	CYED0103
	4 30X, 18 HEOL KEFF	CYED0 104
	5 30X, 18HNUMBER OF BPS, F7.0 /	CYED0105
	6 30X, 18HLOADING	CYED0 106
104	FOR MAT (1H0, 37 X, 10HBATCH DATA / 30X, 1518)	CYED0107
	5 FORMAT (10X, 10X, 10HENRICHMENT, 15F8.3)	CYED0108

106 FORMAT (10X, 10X, 10HFUEL TYPE , 1518)	CYED0 109
107 FORMAT (10X,10X,10HBP FRAC. ,15F8.3)	CYED0110
108 FORMAT (10X, 10X, 10HASSEMBLIES, 1518)	CYED0111
109 FORMAT (20X, 10HDISCHARGED, 1518)	CYED0112
110 FORMAT (20X, 10HSOURCE-C, R, 1518)	CYED0113
111 FORMAT (20X, 10HREGION ID, 15 (4X, A4))	CYED0114
112 FORMAT (20X, 10HBOC BURNUP, 15F8.3)	CYED0 115
113 FORMAT (20X, 10 HEOC BURNUP, 15F8.3)	CYED0116
114 FORMAT (1H0, 37X, 14HDEPLETION DATA /10X, 10HBOS BURNUP, 4X, 5HK EFF ,	CYED0 117
1 1X, 1518)	CYED0118
115 FORMAT (10X, F10.3, F10.4, 8F8.3)	CYED0119
116 FORMAT (1H0, 19X, 9HEDGE DATA / 9X, 6HREGION, 2X, 6HAVAIL., 5X, 5HTOTAL,	CYED0120
1 1X, 8HEXTERIOR, 1X, 1516)	CYED0121
117 FORMAT (2X, 3110, 18, 3X, 1516)	CYED0122
C	CYED0123
END	CYED0124

С	SUBROUTINE SOLVE (A, B, EIGEN, N)	SOLV0001 SOLV0002
	COMMON /TABL1/ NBU, NRICH, BURNUP(20), ENRICH(20), ITYPE(20), BIAS(20),	SOLV0002
	1 BPWRTH (20), CYBIAS	SOLV0003
	CUNMON /10/ 105,106,107	SOLV0005
	COMMON /TABL2/ KINF (20,20), ISOTOP (20,20,6), POWFIS (20)	SOLV0006
	REAL KINF, ISOTOP	SOLV0007
	COMMON /BASIC/ TITLE (20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT,	SOLV0008
	1 BUDIF, ISO, W, A LPHA, ELIM (2), IHAL, POWER, DEBUG	SOLV0009
	LOGICAL DEBUG	SOLV0010
	COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21),	SOLV0011
	1 NFEED (21), NREG (21), CYLOAD (21), CAPFAC (21), CYTIM (21),	SOLV0012
	2 SDLEN (21), IFEDGE (21)	SOLV0013
	COMMON /RGDAT/ NEDGE (2, 1000), IEDGE (21), NAREG (15, 21), JSHUF (15, 21),	SOLV0014
	1 BPFRAC (15, 21), BOCBU (15, 21), EOCBU (15, 21),	SOLV0015
	2 IDXREG (15, 21), NADIS (15, 21)	SOLV0016
	COMMON /CYCL1/ C1RICH(10),C1TYPE(10),C1ID(10),C1LOAD(10),NAC1(10)	SOL VOO 17
	COMMON /FEED / NAFEED (21, 3), FTYPE (21, 3), FLOAD (21, 3), FRICH (21, 3),	SOLV0018
	1 FID(21,3)	SOLV0019
	INTEGER FTYPE, C1TYPE	SOLV0020
	COMMON /COST / IFCOST	SOLV0021
	COMMON /COEF / COFIT (11), TEMP (2), BWRTH, DPWRTH, XEWRTH	SOLV0022
	REAL*8 A (15,15), B (15), BN (15), EIGEN, SUM, EPS, EIGL, DIF	SOLV0023
CDC	REAL A(15,15), B(15), BN(15), EIGEN, SUM, EPS, EIGL, DIF	SOLV0024
С		SOLV0025
0	DATA MAXIT, EPS, OMEGA / 50, 0.00001, 1.80 /, ESP / .001 /	SOLV0026
С		SOLV0027
	SUM=0.0	SOLV0028
	DO 4 I=1, N	SOLV0029
	4 SUM=SUM+B(I) DO 5 I=1,N	SOLV 0030
		SOLV0031
	IF $(SUM.EQ.0.0)$ B $(I) = 1.0/N$ IF $(SUM.GT.0.0)$ B $(I) = B(I)/SUM$	SOLV0032
С	IF (JEBUG) WRITE (IO6, 102) (A(I,J), J=1,N), B(I)	SOLVOO33
C	$\frac{11}{5} \text{ (DEB06) WRITE (100, 102) (R(1,5), 5-1, N), B(1)}{5 \text{ CONTINUE}}$	SOLV0034
	EIGEN=0.0	SOLV0035 SOLV0036
		2014 0020

		SOLV0037
	DO 50 ITER=1, MAXIT DO 30 I=1, N	SOLV0037
	BN(I) = 0.0	SOLV0039
		SOLV0039
	IF (I.EQ.1) GO TO 15	
		SOLV0041
10	DO 10 $J=1,1I$	SOLV0042
	BN(I) = A(I, J) * B(J) + BN(I)	SOLV0043
15	CONTINUE	SOLVO044
	DO 20 $J=I, N$	SOLV0045
	BN(I) = BN(I) + A(I, J) + B(J)	SOLV0046
	CONTINUE	SOLV0047
30	CONTINUE	SOLVO048
	SUM=0.0	SOLV0049
	DO 35 I=1,N	SOLV0050
35	SUM=SUM+BN (I)	SOLV0051
	DIF=0.0	SOLV0052
	DO 40 $I=1, N$	SOLV0053
	BN(I) = BN(I) / SUM	SOLV0054
	DIFF=DMAX1(DIF, DABS(BN(I)-B(I)))	SOLV0055
CDC	DIFF=AMAX1(DIF, ABS(BN(I)-B(I)))	SOLV0056
40	CONTINUE	SOLV0057
	IF (DIF.LE.ESP.AND.DABS(EIGEN-SUM).LE.EPS) GO TO 60	SOLV0058
CDC	IF (DIF.LE.ESP.AND. ABS (EIGEN-SUM).LE.EPS) GO TO 60	SOLV 0059
	EIGEN=SUM	SOLV0060
	DO 45 I=1,N	SOLV0061
	IF (N.LE.5) GO TO 42	SOLV0062
41	B(I) = B(I) + OMEGA*(BN(I) - B(I))	SOLV0063
	GO TO 45	SOLV0064
42	B(I) = BN(I)	SOLV0065
45	CONTINUE	SOLV0066
50	CONTINUE	SOLV0067
6Ŭ	CONTINUE	SOLV0068
	IF (DEBUG) WRITE (106,101) ITER,EIGEN	SOLV0069
	RETURN	SOLV0070
С		SOLV0071
	FORMAT (10X, 5HITER=, 13, 3H K=, F9.5)	SOLV0072
-		50110012

٠

e

:

102 C	FORMAT (10X, 10£12.5)		SOLV0073 SOLV0074
	END	4	SOLV0075

,

•

•

z

.

.

RGED0001 SUBROUTINE RGEDIT **RGED0002** RGED0003 SUBROUTINE RGEDIT - EDITS OUT REGION DISCHARGE SUMMARY RG EL0004 COMMON /TABL1/ NBU, NRICH, BURNUP (20), ENRICH (20), ITYPE (20), BIAS (20), **RGED0005 RGED0006** 1 BPWRTH (20), CYBIAS RGED0007 COMMON /10/ 105,106,107 COMMON /TABL2/ KINF (20,20), ISOTOP (20,20,6), POWFIS (20) RGED0008 **RGED0009** REAL KINF, ISOTOP COMMON /BASIC/ TITLE (20), NASS, ANASS, NADIAM, NALPHA, ULOAD, HEIGHT, **RGED0010 RGED0011** . BUDIF, ISO, W, ALPHA, ELIM (2), IHAL, POWER, DEBUG 1 RGED0012 LOGICAL DEBUG RGED0013 COMMON /CYDAT/ ISURCH(21), ISHUF(21), BP(21), CYBU(21), EOLK(21), NFEED (21), NREG (21), CYLOAD (21), CAPFAC (21), CYTIM (21), **RGED0014** 1 SDLEN(21), IFEDGE(21) RGED0015 2 COMMON /RGDAT/ NEDGE(2,1000), IEDGE(21), NAREG(15,21), JSHUF(15,21), **RGED0016** RGED0017 1 BPFRAC (15, 21), BOCBU (15, 21), EOCBU (15, 21), 2 IDXREG (15,21), NADIS (15,21) **RGED0018** COMMON / CYCL1/ C1RICH(10), C1TYPE(10), C1ID(10), C1LOAD(10), NAC1(10)**RGED0019** COMMON /FEED / NAFEED (21,3), FTYPE (21,3), FLOAD (21,3), FRICH (21,3), RGED0020 **RGED0021** FID (21,3) 1 RGED0022 INTEGER FTYPE, C1TYPE RGED0023 COMMON /COST / IFCOST COMMON /COEF / COFIT(11), TEMP(2), BWRTH, DPWRTH, XEWRTH RGED0024 **RGED0025** DIMENSION BBU (10), NAB (10), ICB (10) **RGED0026** DIMENSION UBU (10), EBU (10), PUBU (10), TBU (10)**RGED0027 RGED0028** NLINE=56DO 50 IC=1,21RGED0029 IF (ISURCH(IC).EQ.0) GO TO 50 RGED0030 RGED0031 NF=NFEED(IC) IF (IC.EQ.1) NF=NREG(1)**RGED0032** IF (NF.EQ.0) GO TO 50 RGED0033 DO 45 IF=1,NF**RGED0034 RG ED0035** NB=0**RGED0036** IF (1C.GT. 1) NAF=NAFEED(IC, IP)

C

C

С

С

	IF (IC.EQ.1) NAF=NAREG (IF, IC)	RG ED0037
	NAF 1= NAF	RGED0038
	IF (IC.EQ.1) GO TO 15	RGED0039
	FRCH = FRICH(IC, IF)	RGED0040
	ITYP=FTYPE (IC, IF)	RGED0041
	RID=FID(IC,IF)	RGED0042
	GO TO 20	RGED0043
15	CONTINUE	RGED0044
	FRCH=C1RICH(IF)	RGED0045
	ITYP=C1TYPE(IF)	RGED0046
	RID=C1ID(IF) .	RGED0047
20	CONTINUE	RGED0048
	B SUM= 0.0	RGED0049
	1DX=100*IC+IF	RGED0050
	DO 10 JC=IC,21	RGED0051
	1F (ISURCH(JC).EQ.0) GO TO 10	RGED0052
	NR = NR EG (JC)	RGED0053
	DO 5 JR=1, NR	RGED0054
	IF (IDXREG(JR, JC).NE.IDX) GO TO 5	RG ED0055
	IF (NADIS(JR, JC) . EQ. 0) GO TO 5	RGED0056
	FRAC=NADIS (JR, JC)	RG ED0057
	NAF 1= NAF 1- NADIS (JR, JC)	RGED0058
	NB = NB + 1	RGED0059
	NB=MINO(NB, 10)	RGED0060
	ICB(NB) = JC	RGED0061
	NAB (NB) = NADIS (JR, JC)	RGED0062
	PRAC=FRAC/NAF	RGED0063
	DISBU=EOCBU (JR, JC) + BUDIF* (1.0 - FRAC)	RGED0064
	BBU(NB) = DISBU	RGED0065
	BSUM=DISBU*NAB (NB) + BSUM	RGED0066
	UBU (NB) = TERP (DISBU, FECH, ITYP, ISOTOP (1,1,1), 0, KB, BP)	RGED0067
	EBU (NB) = TERP (DISBU, FRCH, ITY P, ISOTOP $(1, 1, 2)$, 1, KB, BP)	RGED0068
	PUBU (NB) = TERP (DISBU, FRCH, ITYP, 1SOTOP $(1, 1, 3)$, 1, KB, BP)	RGED0069
	TBU(NB) = 0.0	RGED0070
	IF (IC.EQ.21) GO TO 65	RGED0071
	DO 60 1=IC,JC	RGED0072

ï

60 TBU (NB)=TBU (NB)+CYTIM (I)+SDLEN(I) RGED0073 GO TO 75 RGED0074 GO TO 75 RGED0075 N=0 RGED0075 N=0 RGED0077 CONTINUE RGED0077 JK=JK RGED0077 CONTINUE RGED0078 JK=JSHUF(K,21) RGED0079 K=N4 RGED0070 K=N00(JK,100) RGED0070 TF (JK.GT.K) GO TO 70 RGED0080 TBU (NB)=N*(SDLEN(21)+CYTIM(21)) RGED0085 CONTINUE RGED0085 CONTINUE RGED0080 TF (NB+1+NLINE_LE.55) GO TO 25 RGED0085 NLINS=4 RGED0087 WRITE (L06,101) RGED0081 NLINS=4 RGED0091 SCUNTINUE RGED0092 WRITE (L06,102) RID_IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0093 NLINS=4 RGED0091 NLINS=4 RGED0092 WRITE (L06,102) RID_IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0093 NLINS=4 RGED0095 IF (AB-LE.1) GO TO 35 RGED0096 NULINS=NLINLEANENB*1 RGED0097		•	
GO TO 75 RGED0075 65 CONTINUE RGED0076 N=0 RGED0076 K=Jk RGED0076 70 CONTINUE RGED0076 JK=JSHUP(K,21) RGED0078 M=N+1 RGED0078 K=JMU(JK,100) RGED0070 TP (JK,GT,K) GO TO 70 RGED0081 TF (JK,GT,K) GO TO 70 RGED0083 TS CONTINUE RGED0083 5 CONTINUE RGED0084 6 COUTINUE RGED0084 7 CALL HEAD RGED0084 WRITE (I06,101) RGED0084 NLINZ=4 RGED0091 SUM=SSUM/NAP RGED0093 MILINZ=4 RGED0091 SUM=SSUM/NAP RGED0091 MILINZ=NLINZ=KB+1 RGED0091 1 , TBU(1), UBU(1), ŁBU(1), PUBU(1) RGED0093 1 , TBU(1), UBU(1), ŁBU(1), PUBU(1) RGED0096 1 , TBU(1), UBU(1), LBU(1), PUBU(1), PUBU(1) RGED0097 1 , PUBU(1B) RGED0098 </td <td>60</td> <td>TBU (NB) =TBU (NB) +CYTIM (I) + SDLEN (I)</td> <td>RG E D 0 0 7 3</td>	60	TBU (NB) =TBU (NB) +CYTIM (I) + SDLEN (I)	RG E D 0 0 7 3
N=0 RGED076 K=Jx RGED0776 70 CONTINUE RGED0776 JK=JSHUF(K,21) RGED078 N=N+1 RGED0070 K=MU(JK,100) RGED0080 IF (JK.GT.K) GO TO 70 RGED0081 TBU (NB)=N*(SDLEN(21)+CYTIM(21)) RGED0083 SCONTINUE RGED0084 5 CONTINUE RGED0084 5 CONTINUE RGED0084 5 CONTINUE RGED0084 5 CONTINUE RGED0084 7 (IG6,101) RGED0086 NFTE (IO6,101) RGED0087 RGED0092 RGED0098 WRITE (IO6,101) RGED0090 SCONTINUE RGED0091 BSUM=SUM/NAP RGED0092 wRITE (IO6,102) RID_IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0093 1 SUBU(1),UBU(1),PUBU(1) RGED0092 wRITE (IO6,103) ICB (IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0096 D 30 IB=2,NB RGED0095 1 PUBU(IB) RGED0096 RGED0097 1 PUBU(IB) RGED0098 RGED0098 1 PUBU(IB) RGED0101 RGED0098			RGED0074
No. RGED0077 70 CONTINUE RGED0078 JK=JKUP (K,21) RGED0079 N=N+1 RGED0079 N=N+1 RGED0080 K=MOU(JK,100) RGED0081 IF (JK.GT.K) GO TO 70 RGED0083 TBU (NB)=N* (SDLEN (21) + CYTIM (21)) RGED0084 5 CONTINUE RGED0085 10 CONTINUE RGED0084 5 CONTINUE RGED0086 10 CONTINUE RGED0086 11 RGED0086 RGED0086 12 CONTINUE RGED0087 14 RGED0087 RGED0089 15 CONTINUE RGED0090 15 CONTINUE RGED0091 16 RGED0091 RGED0092 11 TBU (1), JBU (1), FJC, FRCH, ITYP, NAF, BSUM, ICB (1), NAB (1), BBU (1) RGED0092 11 TBU (1), JBU (1), PUBU (1) RGED0091 RGED0092 12 ROMINES RGED0095 RGED0095 RGED0095 14 ROB.LE.1) GO TO 35 <t< td=""><td>65</td><td>CONTINUE</td><td>RGED0075</td></t<>	65	CONTINUE	RGED0075
70 CONTINUE RGED0078 JK=JSHUP(K,21) RGED0079 N=N+1 RGED0079 K=MUD(JK,100) RGED0080 IF (JK,GT.K) GO TO 70 RGED0081 TF (JK,GT.K) GO TO 70 RGED0083 TS CONTINUE RGED0084 5 CONTINUE RGED0084 5 CONTINUE RGED0086 1F (NB+1+NLINE-LE.55) GO TO 25 RGED0087 CALL HEAD RGED0087 CALL HEAD RGED0087 CALL HEAD RGED0090 MILNE=4 RGED0091 RGED0091 RGED0092 VKITE (I06,101) RGED0093 NLINE=NLINE+NB+1 RGED0091 BSUM=BSUM/NAP RGED0092 WRITE (I06,102) RID_IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0093 1 / TBU(1),UBU(1),PUBU(1) RGED0095 JN INE=NBINE+NB+1 RGED0097 NKITE (I06,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0097 30 WRITE (I06,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0097 30 WRITE (I06,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0097 30 WRITE (I06,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0100		N = O	RGED0076
JK=JSHDF(K,21) RGED0079 N=N+1 RGED0080 K=N0b(JK,100) RGED0080 IF (JK.GT.K) GO TO 70 RGED0082 TBU(NB)=N*(SDLEN(21)+CYTIM(21)) RGED0083 75 CONTINUE RGED0084 5 CONTINUE RGED0085 10 CONTINUE RGED0086 11 F (JK.GT.K) GO TO 70 RGED0085 12 CONTINUE RGED0086 13 CONTINUE RGED0086 14 CONTINUE RGED0086 15 CONTINUE RGED0086 16 CONTINUE RGED0086 17 (NB+1+NLINE-LE.55) GO TO 25 RGED0087 18 CLU HEAD RGED0088 WRITE (I06,101) RGED0090 19 NLINE=4 RGED0090 20 CONTINUE RGED0091 10 JTBU(1), UBU(1), EBU(1), PUBU(1) RGED0092 10 TBU(1), UBU(1), EBU(1), PUBU(1) RGED0093 11 TBU(1), UBU(1), EBU(1), PUBU(1) RGED0095 12 JTBU(1), UBU(1), EBU(1), PUBU(1) RGED0096 13 TBU(1), UBU(1), EBU(1), PUBU(1), FBU(1B), UBU(1B), EBU(1B), RGED0096 RGED0096 14 JTBU(1D) RGED0096 RGED0097 30 WRITE (IO6,103) ICB (I		K=J k	RGED0077
JK=JSHUP (K,21) RGED0079 N=N+1 RGED0080 K=A0D (JK,100) RGED0082 TBU (NB)=N* (SDLEN (21)+CYTIN (21)) RGED0083 75 CONTINUE RGED0084 5 CONTINUE RGED0085 10 CONTINUE RGED0086 11 F (JK.GT.K) GO TO 70 RGED0083 75 CONTINUE RGED0084 5 CONTINUE RGED0085 10 CONTINUE RGED0086 11 F (JK.GT.K) GO TO 25 RGED0087 CALL HEAD RGED0088 WRITE (I06,101) RGED0089 NLINE=4 RGED0090 25 CONTINUE RGED0091 BSUM=BSUM/NAF RGED0092 wkITE (I06,102) RID,IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0092 wkITE (I06,103) ICB (IB),IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0093 1TBU(1),UBU(1),ŁBU(1),PUBU(1) RGED0095 RGED0091 NLINE=NLINE+NE1 RGED0092 RGED0093 10	70	CONTINUE	RG E D 0 0 7 8
N=N+1 RGED0080 K=MOU(JK,100) RGED0081 IF (JK.GT.K) GO TO 70 RGED0083 T6U(NB)=N*(SDLEN(21)+CYTIM(21)) RGED0083 75 CONTINUE RGED0083 75 CONTINUE RGED0083 76 CALL HEAD RGED0086 WRITE (IO6,101) RGED0087 CALL HEAD RGED0089 NLINE=4 RGED0092 SUM=BSUM/NAF RGED0091 BSUM=BSUM/NAF RGED0091 MLINE=4 RGED0091 DSUM=BSUM/NAF RGED0091 MLINE=4 RGED0091 SUM=BSUM/NAF RGED0091 MLINE=NLINE+NB+1 RGED0092 J ,TBU(1), UBU(1), PUBU(1) NLINE=NLINE+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0097 JO 0 IB=2,NB RGED0097 30 WRIT2 (IO6,103) ICB (IB), NAB(IB), BBU(IB), TBU (IB), UBU (IB), EBU (IB), RGED0098 1 PUBU(IB) 35 CONTINUE RGED0100 45 CONTINUE RGED0100 95 CONTINUE RGED0100 10 CONTINUE RGED0100 11 PUBU(IB) RGED0102		JK=JSHUF(K,21)	RGED0079
IF (JK.GT.K) GO TO 70 RGED0082 TBU (Mb) = N* (SDLEN (21) + CYTIM (21)) RGED0083 75 CONTINUE RGED0084 5 CONTINUE RGED0085 10 CONTINUE RGED0086 17 CALL HEAD RGED0087 CALL HEAD RGED0088 RGED0089 NLINE=4 RGED0091 RGED0091 SCONTINUE RGED0092 WHITE (I06,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU(1) RGED0092 WHITE (I06,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU(1) RGED0093 NLINE=NLINE+NB+1 RGED0094 NLINE=NLINE+NB+1 RGED0095 IF (AB.LE.1) GO TO 35 RGED0096 DO 30 IB=2, NB RGED0097 30 WRIT2 (I06,103) ICB (IB), NAB(IB), BBU(IB), TBU (IB), UBU (IB), EBU (IB), RGED0098 RGED0098 1 PUBU (IB) RGED0098 35 CONTINUE RGED0091 RGED0098 1 PUBU (IB) RGED0102 RGED0102 RGED0102 RGED0102 RGED0102 RGED0102 RGED0102 NETUNN RGED0102 RGED0104 101			RGED0080
IF (JK.GT.K) GO TO 70 RG ED0082 TBU (NB) = N* (SDLEN (21) + CYTIM (21)) RGED0083 75 CONTINUE RGED0084 5 CONTINUE RGED0085 10 CONTINUE RGED0086 IF (NB+1+NLINE_LE.55) GO TO 25 RGED0087 CALL HEAD RGED0088 WRITE (I06,101) RGED0089 NLINZ=4 RGED0091 ESUM-SUM/NAF RGED0092 WKITE (I06,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU(1) RGED0093 1 , TBU (1), UBU (1), EBU (1), PUBU (1) RGED0094 NLINZ=NLINZ+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0096 DO 30 I B=2, NB RGED0097 30 WRIT2 (I06, 103) ICB (IB), NAB(IB), BBU (IB), TBU (IB), UBU (IB), EBU (IB), RGED0098 RGED0098 1 PUBU (IB) RGED0100 RGED0102 45 CONTINUE RGED0101 RGED0102 10 PORNAT (46X, 14HR2GION SUMMARY //10X, 6HREGION, 2X, 13HPEED IN CYCLE, RGED0102 RGED0103 RGED0104 RGED0105 RGED0104 1 2X, 20 HENKICH TYPE ASMBLY / 2X, 6HDIS BU, 3X, 5HCYCLE, 2X, RGED0105 RGED0105 2 6HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X, RGED0107 RGED0107 <td></td> <td>K=MOD (JK, 100)</td> <td>RGED0081</td>		K=MOD (JK, 100)	RGED0081
TBU (NB) = N* (SDLEN (21) + CYTIM (21)) R GED0083 75 CONTINUE R GED0084 5 CONTINUE R GED0085 10 CONTINUE R GED0086 IF (Nb+1+NLINE.LE.55) GO TO 25 R GED0087 CALL HEAD R GED0088 WRITE (IO6,101) R GED0089 NLINE=4 R GED0091 E SUM=BSUM/NAF R GED0092 WKITE (IO6,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU (1) R GED0093 1 , TBU (1), UBU (1), EBU (1), PUBU (1) R GED0095 1 , TBU (1), UBU (1), EBU (1), PUBU (1) R GED0096 DO 30 IB=2, NB R GED0097 R GED0097 30 WRIT2 (IO6,103) ICB (IB), NAB(IE), BBU (IB), TBU (IB), UBU (IB), EBU (IB), R GED0097 R GED0097 30 WRIT2 (IO6,103) ICB (IB), NAB(IE), BBU (IB), TBU (IB), UBU (IB), EBU (IB), R GED0100 R GED0100 45 CONTINUE R GED0100 R GED0100 45 CONTINUE R GED0101 R GED0102 NETURN R GED0102 R GED0102 1 2X, 20 HENKICH TYPE ASMBLY, 2X, 6HDIS BU, 3X, 5HCYCLE, 2X, R GED0104 101 FOHNAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, R GED0104 1 2X, 2			RGED0082
75 CONTINUE RGED0084 5 CONTINUE RGED0085 10 CONTINUE RGED0086 1F (NB+1+NLINE_LE.55) GO TO 25 RGED0087 CALL HEAD RGED0088 wRITE (106,101) RGED0090 25 CONTINUE RGED0091 BSUM=BSUM/NAF RGED0091 wkITE (106,102) RID,IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0092 wkITE (106,102) RID,IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0093 1 ,TBU(1),UBU(1),EBU(1),PUBU(1) RGED0094 NLINE=NLINE+NE+1 RGED0095 RGED0096 1 ,TBU(1),UBU(1),EBU(1),PUBU(1) RGED0096 00 30 IB=2,NB RGED0097 RGED0097 30 WRITE (106,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0098 RGED0097 31 PUBU(IB) RGED0098 RGED0097 35 CONTINUE RGED0100 RGED0099 RGED0101 35 CONTINUE RGED0102 RGED0101 36 CONTINUE RGED0102 RGED0101 30 WRITE (106,103) ICB(IB),NAB(IB),BBU(IB),SBU(IB),SBU(IB),EBU(IB), RGED0101 RGED0102 36 CONTINUE RGED0100 RGED0101 RGED0101 </td <td></td> <td>• •</td> <td>RGED0083</td>		• •	RGED0083
10CONTINUERGED086IF (NB+1+NLINE.LE.55) GO TO 25RGED087CALL HEADRGED0087CALL HEADRGED0088WRITE (IO6.101)RGED0090DSUM=BSUM/NAPRGED0091WRITE (IO6.102) RID_IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1)RGED0092VRITE (IO6.102) RID_IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1)RGED00931,TBU(1),UBU(1),EBU(1),PUBU(1)RGED0094NLINE=NLINE+NB+1RGED0095IF (NB.LE.1) GO TO 35RGED0096DO 30 IB=2,NBRGED009730 WRITE (IO6,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB),RGED00981PUBU(IB)RGED00965CONTINUERGED00981PUBU(IB)RGED00981PUBU(IB)RGED009935CONTINUERGED010045CONTINUERGED010045CONTINUERGED010150CONTINUERGED0102NETURNRGED0103101FORNAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,RGED0104101FORNAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,RGED010512X,20 EENRICH TYPE ASMELY,2X,6HDIS BU,3X,5HCYCLE,2X,RGED010526HASMELY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X,RGED0107	75		RGED0084
IF (NB+1+NLINE_LE.55) GO TO 25 RGED0087 CALL HEAD RGED0088 WRITE (IO6,101) RGED0089 NLINE=4 RGED0091 25 CONTINUE RGED0091 BSUM=BSUM/NAF RGED0092 wkITE (IO6,102) RID,IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0093 1 ,TBU(1),UBU(1),EBU(1),PUBU(1) RGED0094 NLINE=NLINE+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0096 D0 30 IB=2,NB RGED0097 30 WRIT2 (IO6,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0096 RGED0097 30 WRIT2 (IO6,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0098 RGED0099 35 CONTINUE RGED010 RGED0101 50 CONTINUE RGED010 RGED0101 50 CONTINUE RGED0102 RGED0102 RETURN RGED0104 RGED0104 101 FORNAT (48X,14HR2GION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, RGED0104 RGED0104 101 FORNAT (48X,14HR2GION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, RGED0105 RGED0104 101 FORNAT (48X,14HR2GION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, RGED0105 RGED0104 101 FORNAT (48X,14HR2GION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,	5	CONTINUE	RG ED 0085
CALL HEAD RGED000, COLD COLD COLD COLD COLD COLD COLD COLD	10	CONTINUE	RGED0D86
WRITE (106,101) RG ED0089 NLINE=4 RGED0090 25 CONTINUE RG ED0091 BSUM=BSUM/NAF RGED0092 WRITE (106,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU(1) RGED0092 WRITE (106,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU(1) RGED0093 1 , TBU(1), UBU(1), EBU(1), PUBU(1) RGED0094 NLIN E=NLINE+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0096 DO 30 IB=2, NB RGED0097 30 WRITE (106,103) ICB(IB), NAB(IB), BBU(IB), TBU(IB), UBU(IB), EBU(IB), RGED0097 35 CONTINUE RGED0093 1 PUBU(IB) RGED0094 86 ED0098 RGED0100 45 CONTINUE RGED0097 50 CONTINUE RGED0100 45 CONTINUE RGED0100 45 CONTINUE RGED0100 45 CONTINUE RGED0101 50 CONTINUE RGED0102 RGED0103 RGED0103 101 PORNAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HF EED IN CYCLE, RGED0104 101 PORNAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HF EED IN CYCLE, RGED0106 1 2X, 20 HENKICH TYPE ASMBLY, 2X, 6H		IF (NB+1+NLINE.LE.55) GO TO 25	RG ED0087
NLINE=4RGED009025 CONTINUERGED0091BSUM=BSUM/NAFRGED0092wkITE (I06,102) RID,IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1)RGED00931,TBU(1),UBU(1),EBU(1),PUBU(1)RGED0094NLINE=NLINE+NB+1RGED0095IF (NB.LE.1) GO TO 35RGED0096DO 30 IB=2,NBRGED009730 WRIT2 (I06,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB),RGED009730 WRIT2 (I06,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB),RGED00981PUBU(IB)RGED010050 CONTINUERGED010045 CONTINUERGED010150 CONTINUERGED0102kETURNRGED0102101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, KGED010512X,20HENKICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X, RGED010626HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X, RGED0107		CALL HEAD	RGED0088
25 CONTINUE RGED0091 BSUM=BSUM/NAF RGED0092 wkITE (IO6,102) RID,IF,IC,FRCH,ITYP,NAF,BSUM,ICB(1),NAB(1),BBU(1) RGED0093 1 ,TBU(1),UBU(1),EBU(1),PUBU(1) RGED0094 NLINE=NLINE+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0097 30 WRITE (IO6,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0097 30 WRITE (TO6,103) ICB(IB),NAB(IE),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0098 1 PUBU(IB) RGED0098 35 CONTINUE RGED0098 RGED0100 45 CONTINUE RGED0101 RGED0102 KETURN RGED0104 RGED0104 101 PORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, KGED0105 RGED0104 101 PORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, KGED0105 RGED0104 2 6HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X, KGED0107 RGED0107		WRITE (106,101)	RG ED0089
BSUM=BSUM/NAF RGED0092 WRITE (IO6,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU(1) RGED0093 1 ,TBU(1), UBU(1), EBU(1), PUBU(1) RGED0094 NLINE=NLINE+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0097 30 WRIT2 (IO6,103) ICB(IB), NAB(IB), BBU(IB), TBU(IB), UBU(IB), EBU(IB), RGED0097 30 WRIT2 (OF, 103) ICB(IB), NAB(IB), BBU(IB), TBU(IB), UBU(IB), EBU(IB), RGED0098 1 PUBU(IB) RGED00100 35 CONTINUE RGED0100 45 CONTINUE RGED0102 RGED0102 RGED0103 RGED0103 RGED0104 101 PORNAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, RGED0105 RGED0104 101 PORNAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, RGED0105 RGED0106 2 6HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X, RGED0107		NLINE=4	RGED0090
WkITE (I06,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU(1) RGED0093 1 ,TBU(1), UBU(1), EBU(1), PUBU(1) RGED0094 NLINE=NLINE+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0096 DO 30 IB=2, NB RGED0097 30 WRIT2 (I06, 103) ICB (IB), NAB(IB), BBU(IB), TBU (IB), UBU (IB), EBU (IB), RGED0097 35 CONTINUE RGED0098 1 PUBU (IB) RGED0100 35 CONTINUE RGED0101 45 CONTINUE RGED0102 KETURN RGED0103 101 FORNAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, RGED0105 RGED0104 101 FORNAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, 2X, BGED0105 RGED0106 2 6HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X, RGED0107	25	CONTINUE	RGED0091
1 ,TBU(1),UBU(1),EBU(1),PUBU(1) RGED0094 NLINE=NLINE+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0096 DO 30 IB=2,NB RGED0097 30 WRIT2 (I06,103) ICB (IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0098 1 PUBU(IB) 35 CONTINUE RGED0100 45 CONTINUE RGED0100 45 CONTINUE, RGED0101 50 CONTINUE, RGED0102 RGED0102 RGED0103 RGED0103 RGED0104 101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, RGED0105 RGED0105 1 2X,20HENRICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X, RGED0106 2 6HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X, RGED0107		BSUM=BSUM/NAF	RGED0092
NLINE=NLINE+NB+1 RGED0095 IF (NB.LE.1) GO TO 35 RGED0096 DO 30 IB=2,NB RGED0097 30 WRIT2 (I06,103) ICB (IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0098 1 PUBU(IB) 35 CONTINUE RGED0100 45 CONTINUE RGED0101 50 CONTINUE RGED0102 kETURN RGED0103 101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, RGED0105 RGED0104 101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, RGED0105 RGED0105 2 6HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X, RGED0107		WRITE (106,102) RID, IF, IC, FRCH, ITYP, NAF, BSUM, ICB(1), NAB(1), BBU(1)	RGED0093
IF (NB.LE.1) GO TO 35 RGED0096 DO 30 IB=2,NB RGED0097 30 WRIT2 (I06,103) ICB(IB),NAB(IB),BBU(IB),TBU(IB),UBU(IB),EBU(IB), RGED0098 1 PUBU(IB) 35 CONTINUE RGED0100 45 CONTINUE RGED0101 50 CONTINUE, RGED0102 RETURN RGED0102 101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, RGED0104 101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, RGED0105 1 2X,20HENRICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X, RGED0106 2 6HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X, RGED0107		1 , TBU (1) , UBU (1) , EBU (1) , PUBU (1)	RGED0094
DO 30 I B= 2, NB 30 WRIT2 (IO6, 103) ICB (IB), NAB(IB), BBU(IB), TBU(IB), UBU(IB), EBU(IB), 1 PUBU(IB) 35 CONTINUE 45 CONTINUE 50 CONTINUE, RGED0102 RETURN 101 FORMAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, 1 2X, 20HENRICH TYPE ASMBLY, 2X, 6HDIS BU, 3X, 5HCYCLE, 2X, 2 6HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X, RGED0107 RGED0107 RGED0107		NLINE=NLINE+NB+1	RGED0095
30 WRITZ (106,103) ICB (IB), NAB (IB), BBU (IB), TBU (IB), UBU (IB), EBU (IB), RGED0098 1 PUBU (IB) RGED0099 35 CONTINUE RGED0100 45 CONTINUE RGED0101 50 CONTINUE, RGED0102 RETURN RGED0102 101 FORMAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, RGED0104 101 FORMAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, RGED0105 1 2X, 20HENRICH TYPE ASMBLY, 2X, 6HDIS BU, 3X, 5HCYCLE, 2X, RGED0106 2 6HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X, RGED0107		IF (NB.LE.1) GO TO 35	RGED0096
1 PUBU(IB) RGED0099 35 CONTINUE RGED0100 45 CONTINUE RGED0101 50 CONTINUE RGED0102 RETURN RGED0103 RGED0104 101 FORMAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE, RGED0104 RGED0105 1 2X, 20HENRICH TYPE ASMBLY, 2X, 6HDIS BU, 3X, 5HCYCLE, 2X, RGED0106 RGED0106 2 6HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X, RGED0107		DO $30 IB=2, NB$	RGED0097
35 CONTINUERGED010045 CONTINUERGED010150 CONTINUERGED0102RETURNRGED0103RGED0104RGED0104101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,RGED010512X,20HENRICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X,RGED010626HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X,RGED0107	30	WRITZ (106,103) ICB (IB), NAB (IB), BBU (IB), TBU (IB), UBU (IB), EBU (IB),	RG E D 0 0 9 8
45 CONTINUERGED010150 CONTINUERGED0102RETURNRGED0103101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,RGED0104101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,RGED01051 2X,20HENRICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X,RGED01062 6HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X,RGED0107		1 PUBU(IB)	RGED0099
50 CONTINUE, RETURNRGED0102 RGED0103 RGED0104101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE, 1 2X,20HENRICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X, 2 6HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X,RGED0102 RGED0105 RGED0105 RGED0106 RGED0107	35	CONTINUE	RGED0100
RETURNRGED0103101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,RGED0104101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,RGED01051 2X,20HENRICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X,RGED01062 6HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X,RGED0107	45	CONTINUE	RGED0101
RGED0104101 FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,12X,20HENRICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X,26HASMBLY,3X,6HDIS BU,4X,4HTIME,3X,5HDIS U,2X,6HENRICH,2X,RGED0107	50	CONTINUE ,	RGED0102
101 FORMAT (48X, 14HREGION SUMMARY //10X, 6HREGION, 2X, 13HFEED IN CYCLE,RGED010512X, 20 HENRICH TYPE ASMBLY, 2X, 6HDIS BU, 3X, 5HCYCLE, 2X,RGED010626HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X,RGED0107		RETURN	RG ED0103
12X,20HENRICHTYPEASMBLY,2X,6HDISBU,3X,5HCYCLE,2X,RGED010626HASMBLY,3X,6HDISBU,4X,4HTIME,3X,5HDISU,2X,6HENRICH,2X,RGED0107			RGED0104
2 6HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X, RGED0107	101	FORMAT (48X,14HREGION SUMMARY //10X,6HREGION,2X,13HFEED IN CYCLE,	RGED0105
· · · · · · · · · · · · · · · · · · ·		1 2X,20HENRICH TYPE ASMBLY,2X,6HDIS BU,3X,5HCYCLE,2X,	RGED0106
3 6HDIS PU) RGED0108		2 6HASMBLY, 3X, 6HDIS BU, 4X, 4HTIME, 3X, 5HDIS U, 2X, 6HENRICH, 2X,	
		3 6HDIS PU)	RGED0108

٠

÷.

С

.

102 FORMAT (1H0,10X,A4,4X,12,7X,I2,4X,F5.3,3X,I2,4X,I3,4X,F6.3,4X,I2,	RG ED0109
1 6X,I3,4X,F6.3,F8.0,3F8.4)	RGED0110
103 FORMAT (65X,I2,6X,I3,4X,F6.3,F8.0,3F8.4)	RGED0111
C	RGED0112
END	RGED0113

.

1

.