OAK RIDGE NATIONAL LABORATORY LIBRARY

3 4456 0024474 1

ORNL/TM-8362

ornl

OAK
RIDGE
NATIONAL
LABORATORY

UNION
CARBIDE

DOS: The Discrete Ordinates System

W. A. Rhoades
M. B. Emmett

OAK RIDGE NATIONAL LABORATORY
CENTRAL RESEARCH LIBRARY
CIRCULATION SECTION
4500N ROOM 175
LIBRARY LOAN COPY
DO NOT TRANSFER TO ANOTHER PERSON.
If you wish someone else to see this report, send in name with report and the library will arrange a loan.
ORNL-118 (6-97)

OPERATED BY
UNION CARBIDE CORPORATION
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
NTIS price codes—Printed Copy: A05; Microfiche A01

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ORNL/TM-8362
Distribution Category UC-79d

Contract No. W-7405-eng-26

Engineering Physics Division

DOS: THE DISCRETE ORDINATES SYSTEM

W. A. Rhoades
M. B. Emmett*

Manuscript Completed: May 1982

Date Published - September 1982

This Work Sponsored by
The Defense Nuclear Agency and
DOE Office of Energy Technology,
Division of Reactor Research & Technology

*Computer Sciences Division

OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37830
operated by
UNION CARBIDE CORPORATION
for the
DEPARTMENT OF ENERGY

OAK RIDGE NATIONAL LABORATORY LIBRARY
3 4456 0024474 1

PART I - SYSTEM DESCRIPTION

TABLE OF CONTENTS

SECTION 1. DOS - THE DISCRETE ORDINATES SYSTEM 3

1.1. Program Abstract 5
1.2. Background 7
1.3. User Information 7
1.4. Programmer Information 8
1.5. References for DOS System Codes. 13

PART II - MODULE DESCRIPTIONS

TABLE OF CONTENTS

SECTION 1. THE DOS DRIVER PROGRAM. 17

1.1. Program Abstract 19
1.2. Background 21
1.3. User Information 21
1.4. Programmer Information 21

SECTION 2. BNDRYS: BOUNDARY SOURCE FILE MANIPULATION PROGRAM. . . . 25

2.1. Program Abstract 27
2.2. Background 29
2.3. User Information 29
2.4. Programmer Information 31
2.5. References 31

SECTION 3. GIP: GROUP-ORGANIZED CROSS SECTION INPUT PROGRAM 33

3.1. Program Abstract 35
3.2. General Description. 38
3.3. User Information 39
3.4. Programmer Information 46
3.5. References 46

SECTION 4. RTFLUM: A MODULE FOR CONVERTING, EXPANDING, AND EDITING STANDARD DATA FILES 47

4.1. Program Abstract 49
4.2. General Description. 51

SECTION 5. PROGRAMMER INFORMATION FOR DOT SUBSYSTEM. 61

5.1. Inter-Machine Adaptability 63
5.2. Service Subroutines. 63
5.3. References 65

SECTION 6. ABSTRACTS OF MODULES PREVIOUSLY PUBLISHED 67

6.1. An Updated Version of the DOT 4 One- and Two-Dimensional Neutron/Photon Transport Code. 69
6.2. A User's Manual for ANISN, A One-Dimensional Discrete Ordinates Transport Code with Anisotropic Scattering . . . 77
6.3. DOPES - Discrete Ordinates Perturbation System 81
6.4. DOGS - A Collection of Graphics for Support of Discrete-Ordinates Codes 85

PART I

SYSTEM DESCRIPTION

Section 1. DOS: The Discrete Ordinates System

Editors: W. A. Rhoades
M. B. Emmett*

Section Contributors: R. L. Childs*
M. B. Emmett*
R. A. Lillie
W. A. Rhoades
D. B. Simpson**
E. T. Tomlinson***

*Computer Sciences Division
**TVA, Knoxville, TN.
***TVA, Chattanooga, TN.

1.1. Program Abstract

1.1.1. Program: DOS: The Discrete Ordinates System

1.1.2. Problem Solved: The Discrete Ordinates System determines the flux of neutrons or photons due either to fixed sources specified by the user or to sources generated by particle interaction with the problem materials. It also determines numerous secondary results which depend upon flux. Criticality searches can be performed. Numerous input, output, and file manipulation facilities are provided.

1.1.3. Method of Solution: The DOS driver program reads the problem specification from an input file and calls various program modules into execution as specified by the input file.

1.1.4. Related Codes and Materials: DOS includes a control driver and numerous subsystems which can be executed from it:

1. DRIVER Subsystems - Controls execution of member programs
2. DOT Subsystem - 1-D or 2-D transport calculations
3. ANISN Subsystem - 1-D transport calculations
4. DOPES Subsystem - Transport perturbation calculations
5. DOGS Subsystem - Results summaries and plotting

1.1.5. Restrictions: The order of module execution must be completely determined by the input stream in the present configuration. No run-time decision-making or user intervention is provided. The program modules must reside as executable load modules in a PDS.

1.1.6. Computers: While many of the modules are individually operable on various types of computers, the system is operable as a whole only on IBM 370/3033 or compatible computers.

1.1.7. Running Time: DOS problems require from 1 second to many hours on conventional computers, depending upon the problems solved.

1.1.8. Programming Languages: DOS members are programmed in standard FORTRAN, insofar as possible. In several instances, optional assembler-language routines enhance program capability or convenience.

1.1.9. Operating System: No special requirements for these codes except for the DOS driver. The driver presently operates only under IBM MVS/JES2 or compatible systems.

1.1.10. Machine Requirements: Useful problems can be solved in 270 K-bytes of memory on a machine having at least the capability of the IBM 370/3033.

1.1.11. Authors or major contributors: Noted by subsection.

1.1.12. References:

1. W. A. Rhoades and M. B. Emmett, "DOS: The Discrete Ordinates System," ORNL/TM-8362 (August 1982).

1.1.13. Material Available: User's document and source programs are available through RSIC, Box X, Oak Ridge, TN, 37830.

1.1.14. Abstract Prepared By: W. A. Rhoades

1.2. Background

The range and complexity of tasks performed by modern nuclear codes have lead to the adoption of the "code system" concept. To avoid the construction of a single mammoth code to perform such a range of tasks, semi-independent codes are linked together in one way or another. Since these can be developed and tested independently, programming costs are minimized. New "modules," as the system components are called, can be added to the system even though they have not been developed specifically for that use.

The DOS system consists of a family of interrelated codes which solve radiation transport problems, and which can be executed under control of a single driver. Redundant input data specifications are reduced. Interface files which transmit data from code to code are standardized and compatible, insofar as funding and personnel availability permit.

1.3. User Information

1.3.1. Subsystems and Modules

The various modules comprising DOS are grouped into subsystems. In general, each subsystem is operable separately, is distributed independently, and is under separate configuration management. All can be operated as a single unit under the DOS DRIVER. The subsystems are:

1. DRIVER Subsystem - Controls reading and parceling of input data; directs execution of other modules.

2. DOT Subsystem - Includes the DOT 4 1-D and 2-D neutron/photon transport code and several peripheral modules which process input and output for it.

3. ANISN Subsystem - 1-D neutron photon calculations, like DOT 4, but, perhaps, simpler to use.

4. DOPES Subsystem - Performs detailed perturbation calculations based upon DOT output.

5. DOGS Subsystem - Performs summaries and plots of DOT output.

The modules are listed in Table 1.1.

1.3.2. Method of Operation

The general operation of the system is illustrated in Fig. 1.1. The user prepares an input stream as illustrated in Figs. 1.2 and 1.3. The DOS procedure illustrated may be stored in the system procedure library. The data stream consists of control cards containing the symbol "=" and the program module name. The data for each module follows the control card for that module. An "=END" card terminates the input stream. The DOS driver executes the modules in the order indicated.

1.4. Programmer Information

1.4.1. Requirements for Program Module Compatibility

Extensive interface standardization and redundancy avoidance is obviously helpful, but a minimum set of requirements is sufficient to allow modules to be executed together:

1. The program modules must be stored as executable load modules in the data set containing the driver or in a PDS which can be concatenated to the driver PDS.

2. The program modules must finish with a condition code of 0 unless an error has been encountered.

3. Any main storage allocation must be deallocated.

4. Any direct-access data sets must be closed.

5. The following logical unit requirements must be met:

1-4,8	sequential scratch, defined by the procedure
5	card-image input for individual program modules
6,90	printed output stream
7	card-image output
9-80	user-defined (optional)
81-89,91-97	scratch files[1]
98-99	scratch for driver program

6. When a module uses two printout files, it is intended that 6 be the small-scale file and that 90 be a large-scale file for microfiche processing, etc.

1.4.2. Other Capability and Limitations

In the present system, the execution procedure is explicitly determined by the input stream. Since all of the control logic is contained in a FORTRAN subroutine, DOS could easily be extended to allow the execution

Table 1.1. DOS Member Modules

Notes*	Subsystem	Module	Description
1	DRIVER	DRIVER	Initiates execution of modules as directed by CONTROL module.
		CONTROL	Reads input data, parcels problem descriptions for the various modules, signals DRIVER module to initiate execution of each member module as required.
1,2	DOT	DOT 4	Solves radiation transport problems in 1-D or 2-D using discrete ordinates or diffusion theory.
		GIP	Pre-processes cross section input for use by DOT or GIP.
		RTFLUM	Converts between varius flux file formats, especially VARFLUM and ISOTXS. Lists or expands flux files.
		BNDRYS	Reformats internal boundary flux files for use as internal boundary sources.
3	ANISN	ANISN	Solves 1-D transport problems using discrete ordinates or diffusion theory.
1,4	DOPES	VIP	Performs spatial forward-adjoint folding. Allows grouping of intervals into regions.
		TPERT	Evaluates reactivity changes based on VIP results.
		DGRAD	Calculates currents required by VIP using DOT scalar diffusion-theory fluxes as input.
5	DOGS	EGAD	Geometry display.
		ISOPLOT4	Contour plots of 2-D fluxes.
		FORM	Surface plots of 2-D fluxes.
		ACTUAL	Activity calculations.
		TOOTH	Contributon calculation and plotting.
		ASPECT	Energy spectrum plotting.

*Notes:
1. Modules are described in later sections of this document.
2. A fuller treatment of DOT 4 is given in Reference 1.
3. A fuller treatment of ANISN is given in Reference 2.
4. A fuller treatment of DOPES is given in Reference 3.
5. A fuller treatment of DOGS is given in Reference 4.

Fig. 1.1. Operation of DOS Driver.

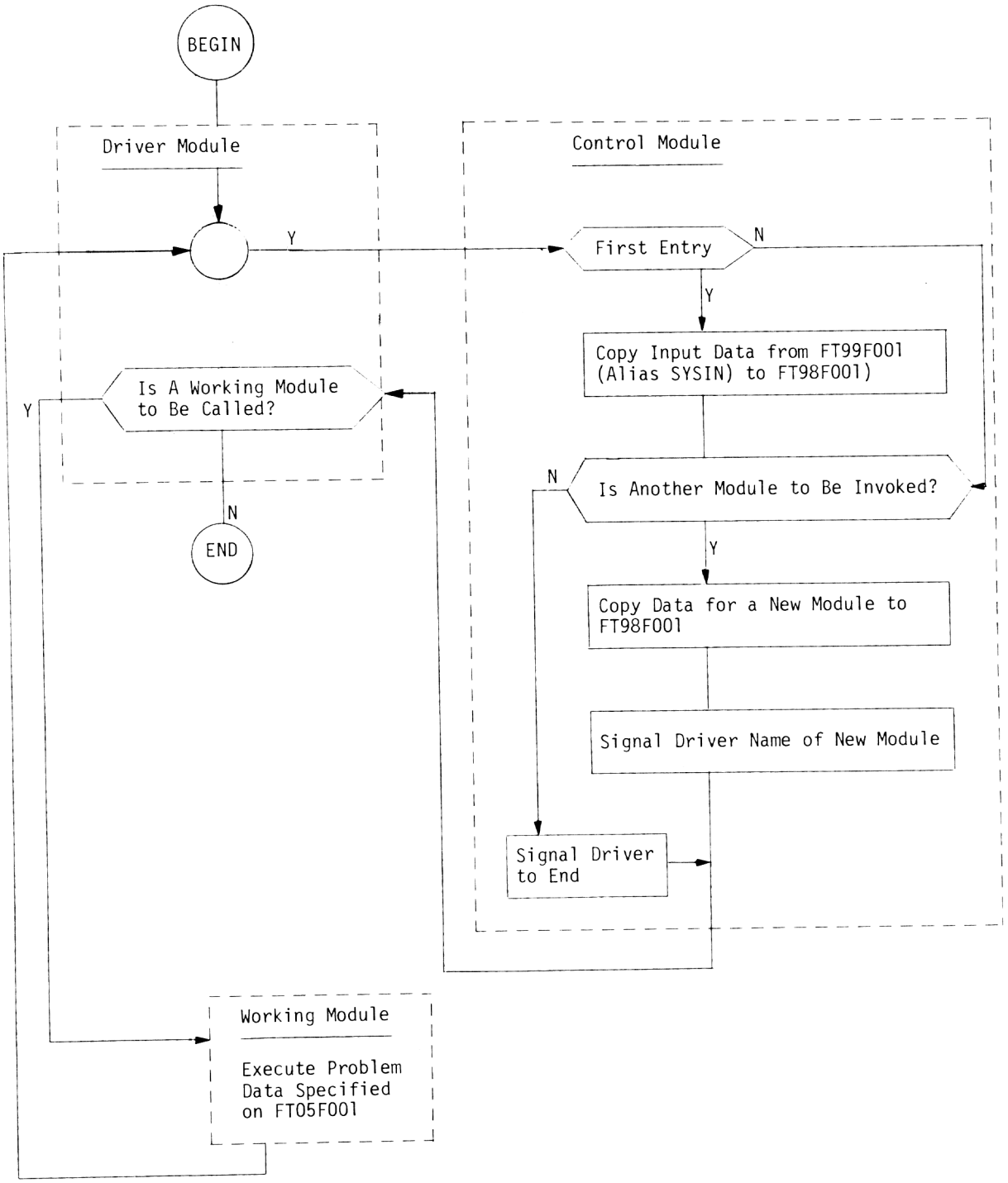

Fig. 1.2. DOS Procedure.

```
//DOS PROC PROG=DRIVER,DSNAM='X.WAF14636.PROG',UNT=,VLM=,
//   PAR=,FT99=SYSTP,SYS=SYSDA,CYL=CYL,REG=384K,BLK=6144,GOTIME=1440,
//   PABLK=300,SABLK=300,DACYL=5,LGCYL=10,MXCYL=20,Q6=A,Q90=A
//* ----------------PREPARED BY W.A.RHOADES, 4-6104-------------------
//GO EXEC PGM=&PROG,TIME=&GOTIME,PARM=&PAR,REGION=&REG
//STEPLIB  DD DSN=&DSNAM,DISP=SHR,UNIT=&UNT,VOL=SER=&VLM
//SYSPRINT DD SYSOUT=&Q90
//FT06F001 DD SYSOUT=&Q6
//FT07F001 DD SYSOUT=B
//FT90F001 DD SYSOUT=&Q90
//FT01F001  DD UNIT=&SYS,DISP=(NEW,DELETE),
//   SPACE=(&BLK,(&PABLK,&SABLK)),
//   DCB=(LRECL=X,BLKSIZE=&BLK,RECFM=VBS,BUFNO=1)
//FT02F001  DD UNIT=&SYS,DISP=(NEW,DELETE),
//   SPACE=(&BLK,(&PABLK,&SABLK)),
//   DCB=(LRECL=X,BLKSIZE=&BLK,RECFM=VBS,BUFNO=1)
//FT03F001  DD UNIT=&SYS,DISP=(NEW,DELETE),
//   SPACE=(&BLK,(&PABLK,&SABLK)),
//   DCB=(LRECL=X,BLKSIZE=&BLK,RECFM=VBS,BUFNO=1)
//FT04F001  DD UNIT=&SYS,DISP=(NEW,DELETE),
//   SPACE=(&BLK,(&PABLK,&SABLK)),
//   DCB=(LRECL=X,BLKSIZE=&BLK,RECFM=VBS,BUFNO=1)
//FT05F001  DD  UNIT=&SYS,SPACE=(80,(50,25)),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200,BUFNO=1)
//FT08F001  DD UNIT=&SYS,DISP=(NEW,PASS),DSN=&&SIGMAS,
//   SPACE=(&BLK,(&PABLK,&SABLK)),
//   DCB=(LRECL=X,BLKSIZE=&BLK,RECFM=VBS,BUFNO=1)
//FT81F001 DD UNIT=&SYS,DISP=(,DELETE),SPACE=(&CYL,&DACYL,,CONTIG)
//FT82F001 DD UNIT=&SYS,DISP=(,DELETE),SPACE=(&CYL,&DACYL,,CONTIG)
//FT83F001 DD UNIT=&SYS,DISP=(,DELETE),SPACE=(&CYL,&DACYL,,CONTIG)
//FT84F001 DD UNIT=&SYS,DISP=(,DELETE),SPACE=(&CYL,&LGCYL,,CONTIG)
//FT91F001 DD UNIT=&SYS,DISP=(,DELETE),SPACE=(&CYL,&MXCYL,,CONTIG)
//FT92F001 DD UNIT=(&SYS,SEP=FT91F001),DISP=(,DELETE),
//   SPACE=(&CYL,&DACYL,,CONTIG)
//FT93F001 DD UNIT=(&SYS,SEP=(FT91F001,FT92F001)),DISP=(,DELETE),
//   SPACE=(&CYL,&DACYL,,CONTIG)
//FT94F001 DD UNIT=&SYS,DISP=(,DELETE),SPACE=(&CYL,&LGCYL,,CONTIG)
//FT95F001 DD UNIT=&SYS,DISP=(,DELETE),SPACE=(&CYL,&DACYL,,CONTIG)
//FT98F001  DD  UNIT=&SYS,SPACE=(80,(50,25)),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200,BUFNO=1)
//FT99F001  DD  DDNAME=&FT99
//* EXAMPLES OF PROC USE - - - - -
//* //DOT EXEC DOS,REG=384K,GOTIME='(8,30)' EXECUTE JOB ON DOS DRIVER
//    PEND
```

Fig. 1.3. Typical Problem Setup.

```
//EXEC    DOS

= GIP

  (GIP DATA)

= DOT

  (DOT DATA)

= END
```

path to be decided based upon previous module results. This would, for example, allow iteration between modules. Funds for such an application have not been identified, however.

The driver subsystem is inherently machine-dependent. The existing driver is operable only on IBM computers. A similar driver could probably be constructed for CDC systems.

1.4.3. Relationship to Other Systems

Subsequent to the development of the DOS driver program, a similar driver was developed for use with the SCALE system,[5] and the modules of the DOS system are operable under the SCALE driver, providing compatible JCL is provided.

1.5. References for DOS System Codes

1. W. A. Rhoades and R. L. Childs, "An Updated Version of the DOT-4 One- and Two-Dimensional Neutron/Photon Transport Code," ORNL-5851 (in publication).

2. W. W. Engle, Jr., "A User's Manual for ANISN -- A One-Dimensional Discrete Ordinates Transport Code with Anisotropic Scattering," K-1693 (March 30, 1967).

3. E. T. Tomlinson, R. L. Childs, and R. A. Lillie, "DOS Perturbation Modules DGRAD/VIP/PERT," ORNL/CSD/TM-116 (May 1980).

4. D. T. Ingersoll and C. O. Slater, "DOGS - A Collection of Graphics for Support of Discrete Ordinates Codes," ORNL/TM-7188 (March 1980).

5. R. M. Westfall et al., "SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluation," ORNL/NUREG/CSD-2 (January 1980).

PART II

MODULE DESCRIPTIONS

Section 1. The DOS Driver Program

D. B. Simpson*
W. A. Rhoades

*TVA, Knoxville, TN.

1.1. Program Abstract

1.1.1. Program: The DOS Driver Program

1.1.2. Problem Solved: The DOS driver program initiates the execution of one or more program modules in a sequence specified in the user-prepared input stream.

1.1.3. Method of Solution: The resident driver initially calls a control module, passing a resident block of control information. The control module scans the input stream to identify the first module to be executed, and to isolate the input data for that module. The resident driver then initiates execution of the indicated program module. The control module is recalled after each program module for determination of the next execution segment.

1.1.4. Related Codes and Materials: A separate document describes the DOS system and its program modules.

1.1.5. Restrictions: A few restrictions on the JCL available to members are listed in the DOS documentation.

1.1.6. Computer: An IBM-compatible system is required for the driver.

1.1.7. Running Time: The driver program, itself, requires much less than a second of CPU time.

1.1.8. Programming Languages: IBM FORTRAN IV and 360/370 assembler language are used.

1.1.9. Operating System: DOS driver presently operates under a standard IBM MVS system.

1.1.10. Machine Requirements: No special requirements are applicable.

1.1.11. Authors or Major Contributors:

D. B. Simpson, TVA, Knoxville, TN

W. A. Rhoades, ORNL, Oak Ridge, TN

L. M. Petrie, Union Carbide Nuclear Division, Oak Ridge, TN

P. K. Erickson

1.1.12. References:

1. W. A. Rhoades and M. B. Emmett, "DOS: The Discrete Ordinates System," ORNL/TM-8362 (August 1982).

1.1.13. Abstract Prepared By: W. A. Rhoades

1.2. Background

The DOS driver program allows several program modules to be executed within a single job step. Data are communicated between a resident "DRIVER" module and the subordinate modules through data files defined in DD statements, and through a small array of control data resident in the driver. Only the driver and this control array are in memory at all times.

1.3. User Information

The execution sequence is controlled by cards containing the symbol "=" and the name of the desired modules. A module called "CONTROL" is executed at the beginning of each DOS job. CONTROL moves data from the input stream (FT99F001) to a scratch file (FT98F001) and edits it. The first program module is determined from the first card, the proper data cards for that module are copied to FT05F001, and the execution is returned to the driver.

The driver initiates the desired program module and, upon completion of the program, recalls the control module. The condition code is checked after each module execution. A non-zero code results in termination of the processing. Upon recall, CONTROL identifies the next program module from the input stream and repeats the process until an "=END" card is encountered. The control module then signals the driver to end the execution.

1.4. Programmer Information

1.4.1. Driver Loading

The driver program consists of two load modules:

1. A resident "DRIVER" program, a single main program written in assembler language, initiates the subordinate modules and terminates execution.

2. A non-resident "CONTROL" program consisting of a main program named "BRAIN" plus a subroutine named "CONTRL" processes all decision making and preparation of input segments for the program modules. The sole function of BRAIN is to link the resident block of control information to CONTROL. BRAIN must be marked as the entry point.

1.4.2. Program Module Loading

Program modules may be loaded as executable load modules into the PDS containing the driver load modules, or they may be contained in a PDS concatenated to the DRIVER PDS in the STEPLIB DD statement.

1.4.3. Use of the Control Array by Other Programs

Other programs may access and use portions of the resident control array not reserved for the driver. Such programs must have BRAIN as their main program. BRAIN will then call a subroutine named CONTRL with the resident array as an argument. The user may construct a subroutine named CONTRL for his program to control subsequent execution. The contents of the resident array are specified in Appendix 1.A.

APPENDIX 1.A. Contents of the Resident Control Array

A(1) = 1st Word of Name of Next Module to Call (Preset to '$$$$')
A(2) = 2nd Word of Name of Next Module to Call (Preset to Length of an Array)
A(3) = Number of Entries to CONTRL
A(4) = Number of Input Cards for Which Processing Has Been Completed
A(5) = 1st Word of Name of Module Following the Next
A(6) = 2nd Word of Name of Module Following the Next
A(7) To A(20) = Reserved for Future Use
A(20) To A(100) = Available

Section 2. BNDRYS: Boundary Source File Manipulation Program

M. B. Emmett*

*Computer Sciences Division.

2.1. Program Abstract

2.1.1. Program: BNDRYS - Internal Boundary Source Code

2.1.2. Problem Solved: BNDRYS reformats an internal boundary flux file provided in the BNDRYS format,[1] producing an internal boundary source file, also in the BNDRYS format, for use in a subsequent calculation.

2.1.3. Method of Solution: BNDRYS selects the appropriate flux sets from a potentially larger collection in the input file. In the case of I-boundary sets, either rightward or leftward fluxes are retained, with the others set to 0. For J-boundary sets, either upward or downward fluxes are retained. The fluxes are multiplied by user-supplied rescaling factors and padded with 0's at both ends as specified. A set of fluxes can also be reflected to make a mirror image. The resulting fluxes are then moved to an output file in BNDRYS format.

BNDRYS allows flux emerging from a reactor core to be determined in an initial eigenvalue problem, and then to be used as a fixed source in a subsequent reflector, vessel, and shield problem. The core can be completely internal to the second problem. For accuracy, the core material is replaced with a highly absorbent material in such a problem.

2.1.4. Related Codes and Materials: The DOT code[1] prepares input files for BNDRYS, and uses the output files.

2.1.5. Restrictions: None.

2.1.6. Computers: Although the BNDRYS codes should be adaptable to many types of computers, only IBM 360/370 and CRAY operation have been demonstrated.

2.1.7. Running Time: BNDRYS problems require only a few seconds of CPU time.

2.1.8. Programming Languages: BNDRYS is programmed in standard FORTRAN, insofar as possible. Certain optional IBM assembler-language routines which enhance program capability or convenience are provided.

2.1.9. Operating System: BNDRYS presently operates under IBM MVS/JES2 or CRAY CTSS.

2.1.10. Machine Requirements: Useful problems can be solved in 270 K-bytes of memory on a machine having at least the capability of the IBM 360/75.

2.1.11. Authors or major contributors:

M. B. Emmett, Union Carbide Nuclear Division, Oak Ridge, TN.

2.1.12. References:

1. W. A. Rhoades et al., "The DOT IV Two-Dimensional Discrete-Ordinates Transport Code with Space-Dependent Mesh and Quadrature," ORNL/TM-6529 (January 1979).

2. W. A. Rhoades and M. B. Emmett, "DOS: The Discrete Ordinates System," ORNL/TM-8362 (August 1982).

2.2. Background

BNDRYS is a computer code which restructures and reformats data for use as an internal boundary condition. Its typical use is in coupling a core eigenvalue solution to a large shield problem.

2.3. User Information

2.3.1. Card-Image Input Specifications

Following the title card, all data are read using the FIDO input system,[1] and familiarity with that system is assumed. Data arrays are entered in blocks, each terminated by a "T." Unused data arrays are not entered, but a T must be entered to signal the termination of each block, in any case. The data ordering is illustrated in Fig. 2.1.

2.3.2. Input File Specification

The input file is an enternal boundary flux file in the BNDRYS format.[1]

2.3.3. Output File Specification

The output file is an internal boundary source file that has been reformatted so that the I-directional source array contains MMA*JM *NINTFX values and the J-directional source array contains MMA*IM *NJNTFX values.

2.3.4. Logical Unit Requirements

Logical unit 71 - default - input BNDRYS file

Logical unit 72 - default - output file

Logical unit 5 - card input

Logical unit 6 - printed output

The first two unit numbers are overridden if NTIBI and NTIBO are defined in the 1$ array.

2.4. Programmer Information

See Section 5.

2.5. References

1. W. A. Rhoades et al., "The DOT IV Two-Dimensional Discrete-Ordinates Transport Code with Space-Dependent Mesh and Quadrature," ORNL/TM-6529 (January 1979).

Fig. 2.1. BNDRYS Input Instructions.

```
Title Card    TITL(12)  12A6

Basic Input:  1$     - Integer data

              NTIBI  - Input unit for fluxes:  Default = 71

              NTIBO  - Output unit for fluxes:  Default = 72

              IEDIT  - Edit Control; 0 = no edit

              JMX    - Initial 0's added on J

              JMO    - Total JM output

              IMX    - Initial 0's added on I

              IMO    - Total IM output

              NBUF   - I/O buffer size in K-bytes:  Default = 60

              IDBLA  - Reflection control; 1 = double the JMO; 0 = don't
                       reflect the JMO

              E

              T

Array Input:  3$     - IKEY(NINTFX) where NINTFX is number of I-boundary
                       sources

              ± N; N is output set #, + indicates save μ > 0; -, save μ < 0

              4$     - JKEY(NJNTFX) where NJNTFX is number of J-boundary
                       sources

              ± N; N is output set #, + indicates save η > 0; -, save η < 0

              5*     - ZFLAG(JM) where JM is number of J-intervals on input

                       file.  Multiplier for FII array[1]

              6*     - RFLAG(IMA) where IMA is number of I-intervals on input

                       file.  Multiplier for FJI array[1]

              T

     81*, 82*, 83* data from DOT must be included here.  This is the
quadrature set including weights, mus and etas.

              T
```

Section 3. GIP: Group-Organized Cross Section Input Program

W. A. Rhoades

3.1. Program Abstract

3.1.1. Program: GIP: Group-Organized Cross Section Input Program

3.1.2. Problem Solved: GIP accepts nuclide-organized microscopic cross section data either from the input stream in card-image format or from a data library prepared by the ALC1 program.[1] GIP prepares a group-organized file of microscopic and/or macroscopic cross sections for use by DOT,[2] ANISN,[3] or related codes. Adjoint cross sections can be prepared. The total upscatter cross section can be inserted into the data format as required by DOT and ANISN.

3.1.3. Method of Solution: A simple sequential file sorting technique is used. Macroscopic data are specified by a "mixing table" similar to that of DOT.

3.1.4. Related Codes and Materials: See references.

Reference 2 gives a complete definition of the output file format.

3.1.5. Restrictions: The sorting procedure is well suited to problems where the total volume of cross section data to be processed is, for example, 3 or 4 times the available data storage space. If very large problems must be solved on small computers, a random-access program may be more efficient.

Complex library formats such as ISOTXS[4] are beyond the scope of GIP.

3.1.6. Computers: GIP has performed well on many types of IBM, Amdahl CDC, CRAY, and UNIVAC computers.

3.1.7. Running Time: Most GIP problems require much less than a minute of IBM 360/91 CPU time.

3.1.8. Programming Languages: GIP can be operated entirely in standard FORTRAN. On IBM computers, optional assembler-language programs provide enhanced convenience.

3.1.9. Operating System: No special requirements.

3.1.10. Machine Requirements: Provisions must be made for:

standard input and output files on 5 and 6
sequential scratch files on logical units 1-4
output file on logical unit 8
input library file, if any, on logical unit 9

3.1.11. Authors or Major Contributors:

W. A. Rhoades, ORNL, Oak Ridge, TN
W. W. Engle, Jr., ORNL, Oak Ridge, TN
B. J. Dray

3.1.12. References:

1. W. A. Rhoades, "The ALC1 Program for Cross Section Management," ORNL/TM-4015 (December 1972).

2. W. A. Rhoades et al., "The DOT IV Two-Dimensional Discrete-Ordinates Transport Code with Space-Dependent Mesh and Quadrature," ORNL/TM-6529 (January 1979).

3. W. W. Engle, Jr., "ANISN, A One-Dimensional Discrete Ordinates Transport Code with Anisotropic Scattering," K-1693 (March 1967).

4. R. Douglas O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," LA-6941-MS (September 1977).

5. W. A. Rhoades and M. B. Emmett, "DOS: The Discrete Ordinates System," ORNL/TM-8362 (August 1982).

3.1.13. Abstract Prepared By: W. A. Rhoades

3.2. General Description

3.2.1. Background

GIP accepts nuclide-organized microscopic cross-section data either from the input stream in card-image format or from a data library prepared by the ALC1 program.[1] Macroscopic cross-section mixtures can be prepared as specified by a mixing table similar to that of DOT.[2] The total upscatter cross section can be calculated and inserted into the set. The result is a "GIP" cross section file as defined in Reference 2. Microscopic and/or macroscopic data can be output to the file and/or printed.

The chief virtues of GIP are its extreme simplicity and its high compatibility with DOT, ANISN,[3] and related codes. It is operable on virtually any type of computer which provides sequential scratch files. It uses the FIDO card-image input processor, and shares many subroutines with DOT. Its major disadvantages are the inability to translate complex input files such as ISOTXS[4] and its inefficiency for very large problems.

GIP treats each component of a Legendre expansion as a separate nuclide. This was seen as a requirement in the 1960's, when the code originated. Although certain features of the mixing table expedite the mixing of PL sets, there is little doubt that the code would be more convenient for present-day problems if all of the components were treated as a unit.

3.2.2. Method Used

The cross sections specified by the input stream are selected, and the number of energy groups which can be processed at one time is decided. The

groups for one corefull are selected as each nuclide is read, and dumped onto a source file. The remaining groups, if any, are written to a holding file. When all nuclides have been located, the data are read from the source file into core and written in group-ordered format onto a sorting file. A new corefull of data are then moved from the holding file to the source file, and then to the sorting file as before. Unprocessed groups, if any, are spilled to a second holding file, which is processed as was the first holding file. This sequence continues until all groups have been processed. It happens that one group of data can be written directly to the sorting file each time that the source file is loaded, and this improves efficiency somewhat. In the final step, the group-organized data are read, mixed, edited, and moved to the output file.

The advantages of this system are that it can process a cross section volume of several corefulls with inconsequential cost, using only sequential files, and that it is easily adapted to new computing environments. Unfortunately, if the cross section volume increases to, for example, 10 or more corefulls, the sorting process becomes inefficient.

This method originated with W. W. Engle, Jr., in a program called TAPEMAKER, which was operated on the IBM 7090 before the days of random access files. The basic procedure has changed little since then.

3.3. User Information

3.3.1. Card-Image Input Specifications

Following the title card, all data are read using the FIDO input system,[2] and familiarity with that system is assumed. Data arrays are entered

in blocks, each terminated by a "T." Unused data arrays (e.g. 10$ if MS=0) are not entered, but a T must be entered to signal the termination of each block, in any case. The data ordering is illustrated in Fig. 3.1.

The input of the 14* array is performed by subroutines which have certain restrictions in addition to those of normal FIDO input:

a. The nuclide spaces are preset to 0.0, so that "E" or "FO" will have the effect of filling the remaining space with 0.0, but "E" and "F" must not be used in the input for the last nuclide.

b. The 14* or 14** designation may precede each nuclide block, but is not required except in the first block.

c. Entries, including operators such as "T," following the last data item for nuclides other than the last, and on the same card as the last item, will be ignored.

d. The T which terminates this block must appear alone in column 3 of a separate card.

The mixing table is prepared as described for the DOT code,[2] and the meaning of items 1-8 of the 1$ array are compatible with DOT. NBUF must allow buffer space for the data sets which are listed below. As with DOT, the integers in the mixing table refer to a continuous array of MTM cross section sets, of which the first MCR+MTP are nuclide data read from card-image input and library file, respectively. All of the sets beyond MCR+MTP are preset to 0 before the mixing table is executed. Reference to eigenvalue modification in the DOT description has no application to GIP mixing.

For most DOT problems, I/O manipulation can be reduced by asking for output of only the mixtures from GIP. They are then referred to in DOT such that material 1 is the first GIP mixture, etc. Any microscopic data to be used in DOT activity edits must be included as a mixture, as must nuclides whose concentrations are to be modified in a search.

Multiple problems can be stacked in succession with a blank card between cases. No input data are saved between cases.

3.3.2. Special Diffusion Theory Option

If the cross section set contains a transport cross section in position N, then setting ISCT=-N causes the total cross section to be replaced by the transport cross section and the self-scatter cross section (position IHS) to be reduced such as to maintain a constant absorption cross section. The resulting cross section set is suitable for use in a P0 DOT calculation, especially one using the diffusion package. Negative entries in the 10$ array must not be used with this option.

3.3.3. Upscatter

If IHS>IHT+1, the upscatter processor calculates the total upscattering cross section from each group and places it in the appropriate table position. No user action is required.

3.3.4. Adjoint

If ITH=1, two major reordering steps are accomplished:

1. The inscatter matrix is transposed, i.e. the table position associated with group g describing scattering from g' to g is changed to describe scattering from g to g'.

2. The ordering of the groups is reversed, i.e. data for group IGM appear first in the output file, followed by IGM-1, etc.

Figure 3.2 shows the results of this reordering.

3.3.5. Input Library Format

The input library format from which MTP nuclides are taken is defined in Appendix 3.A.

3.3.6. Output File

The output file is in GIP format.[2] If IOUT=0, the output includes MCR+MTP microscopic cross sections, plus (MTM-MCR-MTP) mixtures. If IOUT=2, only the mixtures will be output. Printed cross section edits are similarly controlled by IPRT.

3.3.7. Logical Unit Requirements

Logical Unit 1 -
Logical Unit 2 -
Logical Unit 3 -
} Scratch, should be blocked for efficiency
Logical Unit 4 -
Logical Unit 9 - Tape Input
Logical Unit 8 - Tape Output
Logical Unit 6 - Printed Output
Logical Unit 5 - Card Input

Logical unit assignments for the first 6 data files can be modified by altering locations 15-20 of the 1$ array to the new logical unit numbers. The new values will not be listed in the input edit, however.

3.4. Programmer Information

See Section 5.

3.5. References

1. W. A. Rhoades, "The ALC1 Program for Cross Section Management.

2. W. A. Rhoades et al., "The DOT IV Two-Dimensional Discrete-Ordinates Transport Code with Space-Dependent Mesh and Quadrature," ORNL/TM-6529 (January 1979).

3. W. W. Engle, Jr., "ANISN, A One-Dimensional Discrete-Ordinates Transport Code with Anisotropic Scattering," K-1693 (March 1967).

4. R. Douglas O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," LA-6941-MS (September 1977).

Figure. 3.1. Input Data Ordering.

Title Card (72-character description)

Parameter Input

1$ Array

IGM	No. of Groups
IHT	Sigma(T) Position
IHS	Sigma(G--G) Position
IHM	Table Length
MS	Mixing Table Length
MCR	No. of Materials from Cards
MTP	No. of Materials from Lib. Tape
MTM	No. of Materials (MCR+MTP+MIXTURES)
ITH	0=Direct Solution, 1=Adjoint
ISCT	Order of Scattering Expansion
IPRT	0=Print All, 1=No Print, 2=Print Mixtures
IOUT	0=Output All Matls., 1=None, 2=Mixt, Only
IDOT	0=ANISN Output, 1=DOTIII, 2=DOTIV
NBUF	No. K-Bytes Buffer Space (Default=60)

E

T

10$, 11$, 12* arrays - mixing table[a] (MS entries each)

7$ array - mix code[a] (optional - MTM entries)

13$ array - nuclide ID numbers on library file (MTP entries)

T

14* array - cross sections for nuclide 1, ANISN format,[b] all groups

14* array - cross sections for nuclide MCR, ANISN format,[b] all groups

T

---

[a]See Reference 2.

[b]See ALC Cross Section Format in Appendix 3.A.

Fig. 3.2. Adjoint Cross Section Reordering.

(3-group example)

Direct	Adjoint
σ_1^F	σ_3^F
σ_1^A	σ_3^A
$\nu\sigma_1^F$	$\nu\sigma_3^F$
σ_1^T	σ_3^T
$\sigma_{1,3}$	$\sigma_{1,3}$
$\sigma_{1,2}$	$\sigma_{2,3}$
$\sigma_{1,1}$	$\sigma_{3,3}$
0	0
0	0
σ_2^F	σ_2^F
σ_2^A	σ_2^A
$\nu\sigma_2^F$	$\nu\sigma_2^F$
σ_2^T	σ_2^T
0	0
$\sigma_{2,3}$	$\sigma_{1,2}$
$\sigma_{2,2}$	$\sigma_{2,2}$
$\sigma_{2,1}$	$\sigma_{3,2}$
0	0
σ_3^F	σ_1^F
σ_3^A	σ_1^A
$\nu\sigma_3^F$	$\nu\sigma_1^F$
σ_3^T	σ_1^T
0	0
0	0
$\sigma_{3,3}$	$\sigma_{1,1}$
$\sigma_{3,2}$	$\sigma_{2,1}$
$\sigma_{3,1}$	$\sigma_{3,1}$

APPENDIX 3.A: ALC Cross Section Library Format

```
C*********************************************************************
C               PROPOSED 19 MAY 80
C
CF      ALC
CE      TRANSPORT CODE CROSS SECTION DATA
C
C*********************************************************************
C
CD              THE TERMINATION RECORD DEFINES THE LENGTH OF THE
CD              DATA SET AND THE NUMBER OF NUCLIDES
C
CD              THE ID VALUES MUST BE IN ASCENDING ORDER
C
C---------------------------------------------------------------------
CS              FILE STRUCTURE
C
CS      RECORD TYPE                                     PRESENT IF
CS      ------------------------------------------      ----------
CS ***************(REPEAT FOR ALL NUCLIDES)
CS *    CONTROL DATA FOR A NUCLIDE                      ALWAYS
CS *    CROSS SECTION DATA FOR A NUCLIDE                ALWAYS
CS ***************
C
CS      TERMINATION                                     ALWAYS
C
C---------------------------------------------------------------------
C
C
C---------------------------------------------------------------------
CR              CONTROL DATA FOR A NUCLIDE
C
CL      NOG,NCTL,NCC,NCID,(NAME(I),I=1,12)
C
CW      16=NUMBER OF WORDS
C
CD      NOG             NUMBER OF ENERGY GROUPS FOR THIS SET
CD      NCTL            LENGTH OF CROSS SECTION TABLE FOR THIS SET
CD      NCC             =0 SET WAS ADDED TO LIBRARY
CD                      =2 SET REPLACED AN EXISTING SET
CD      NCID            IDENTIFICATION NUMBER
CD      NAME            ARBITRARY DESCRIPTION, A4 FORMAT
C
C---------------------------------------------------------------------
C
C
C---------------------------------------------------------------------
CR              CROSS SECTIONS FOR A NUCLIDE
C
CL      ((CRX(IH,IG),IH=1,NCTL),IG=1,NOG)
C
CW      NCTL*NOG=NUMBER OF WORDS
C
CD      CRX             CROSS SECTION DATA ORDERED AS PER ORNL/TM 6529
CD      NCTL            AS SPECIFIED BY PREVIOUS RECORD
CD      NOG             AS SPECIFIED BY PREVIOUS RECORD
C
C---------------------------------------------------------------------
C
C
C---------------------------------------------------------------------
CR              TERMINATION
C
CL      NOG,NCTL,NCC,NCID,(NAME(I),I=1,12)
C
CW      16=NUMBER OF WORDS
C
CD      NCC             7, ALWAYS
CD      OTHER ITEMS     ARBITRARY
C
C---------------------------------------------------------------------
```

Section 4. RTFLUM: A Module for Converting, Expanding, and Editing Standard Data Files

W. A. Rhoades

4.1. Program Abstract

4.1.1. Program: RTFLUM: A Module for Converting, Expanding, and Editing Standard Data Files.

4.1.2. Problem Solved: RTFLUM provides conversion between flux moment files in the DOTIII,[1] VARFLM,[2] or RTFLUX[3] formats. Scalar flux, flux moments, and/or boundary fluxes can be edited by user option.

4.1.3. Method of Solution: Reformatting is done in-core. Essential data required by the output file but absent from the input file must be supplied in the input stream by the user. Default values of unessential data are supplied by the code.

4.1.4. Related Codes and Materials: See references.

4.1.5. Restrictions: None.

4.1.6. Computers: RTFLUM has performed well on many types of IBM, Amdahl, CDC, and CRAY computers.

4.1.7. Running Time: Most RTFLUM problems require much less than a minute of IBM 360/91 CPU time.

4.1.8. Programming Languages: RTFLUM can be operated entirely in standard FORTRAN. On IBM computers, optional assembler-language programs provide enhanced convenience.

4.1.9. Operating System: No special requirements.

4.1.10. Machine Requirements: Provisions must be made for:

standard input and output files on 5 and 6

input and output files on units specified by the user

4.1.11. Authors or Major Contributors:

W. A. Rhoades, ORNL, Oak Ridge, TN

4.1.12. References:

1. F. R. Mynatt et al., "The DOT III Two-Dimensional Discrete-Ordinates Transport Code," ORNL-TM-4280 (September 1973).

2. W. A. Rhoades et al., "The DOT IV Two-Dimensional Discrete-Ordinates Transport Code with Space-Dependent Mesh and Quadrature," ORNL/TM-6529 (January 1979).

3. R. Douglas O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," LA-6941-MS (September 1977).

4. W. A. Rhoades and M. B. Emmett, "DOS: The Discrete Ordinates System," ORNL/TM-8362 (August 1982).

4.1.13. Abstract Prepared By: W. A. Rhoades.

4.2. General Description

4.2.1. Background

The ERDA/RDD Physics Branch Committee on Computer Code Coordination (CCCC) has specified the RTFLUX file format for expressing double-precision scalar, fixed-mesh flux data (LA-5486-MS). This file is produced by VENTURE and certain other codes. The file contains certain identification information, followed by scalar flux for each group, with the first two space dimensions blocked together.

This file was not suitable for DOT IV application. A new single-precision file format, VARFLM, includes flux moments, accommodates an irregular mesh, and includes boundary directional fluxes required for an accurate restart. It also has enough identification to relate the file uniquely to the job which produced it, and has provision for enough mesh information to allow automatic plotting and processing without reference to geometry files. It is blocked as required by row-oriented codes.

A third format in wide use is the "scalar flux tape" produced by DOT III and used by many related plotting, perturbation, and processing codes (ORNL/TM-4280).

The RTFLUM program offers a simple means of editing these various files and of converting from one format to another. This allows VENTURE data to be used as a DOT IV starting guess, or to be plotted with a DOT III plotting code, for example. In addition, provision is made to expand the number of moments or directions implied, so that a low-order output can be used to

start a high-order input. In solving large, deep-penetration problems, a file may often be left incomplete, due to premature problem termination. RTFLUM can repair such output, also.

4.2.2. Application

RTFLUM requires only input and output file definition. If no output file is desired, a dummy is to be supplied. The input data are supplied in FIDO format as used by DOT IV. Two blocks of data are required. The first contains control arrays 1$$ and 2** as listed in the sample problem output, followed by "T." The second block requires only "T" for present applications.

Certain defaults minimize the input requirements, as indicated in the output listing. If a VARFLM file is used as input, the default values of MM, ISCAT, IGM, IM, JM, ISCAI, MMI, and IGMI will be replaced by data from the file. Default values for data beyond IGMI will suffice for many applications.

If RTFLUX is used, MM and ISCAT are assumed to be 6 and 0.

If a DOT III scalar flux tape is used as input, MM, ISCAT, IGM, IM, and JM must be supplied as input. The default values of ISCAI, MMI, and IGMI will cause the values of ISCAT, MM, and IGM to be used.

If the input and output direction numbers are to be different, enter MM and MMI explicitly. If the moment orders are to differ, enter ISCAT and ISCAI explicitly. If the input file does not contain IGM groups of good data, enter a value for IGMI equal to the number of good groups.

The IEDIT parameter is used as follows:

IEDIT =	0	no edit of output
	1	edit scalar output only
	2	edit full moment expansion
	10, 11, or 12 as 0, 1, or 2, but also edit boundary flux.	

4.2.3. Output

The output from a job converting a VARFLM file to DOT III format is illustrated in Fig. 4.1. A set of self-explanatory input sets in Fig. 4.2 help to illustrate the use of the defaults. The output in Fig. 4.1 results from using the first input set in Fig. 4.2.

Fig. 4.1. Sample Problem Output from RTFLUM Case.

```
 =RTFLUM
 VARFLM TO DOTIII, EDIT BOUNDARY FLUXES
 1$$  2  3  10    A7  0  1    E T
 T
 =END
FIGURE 1  OUTPUT FROM RTFLUM RUN
1DRIVER WILL NEXT CALL MODULE  RTFLUM
1* * * PROBLEM BEGAN ON 05/25/82 (145 ) AT TIME 15 HRS, 33.6 MIN. * * *


0******  RTFLUM (ORNL 20 MAR 82) ******
0     1$ ARRAY     20 ENTRIES READ
0     0T
0VARFLM TO DOTIII, EDIT BOUNDARY FLUXES
0FLUX INPUT FILE VARFLM, USER ID SMALL HEAD C, VERSION     0
0CREATED 052582 BY , CHARGE= CASE=WAR059 TIME=1533.5
0TITLE=SMALL HEAD CLOSURE DEMONSTRATION PROBLEM
0 1 $ $
0NTRTF  =     2          INPUT  LOGICAL UNIT NUMBER (DFLT=71)
 NTFOG  =     3          OUTPUT LOGICAL UNIT NUMBER (DFLT=72)
 IEDIT  =    10          EDIT CONTROL
 MM     =    -1          OUTPUT NUMBER OF DIRECTIONS (DFLT=-1)
 ISCAT  =    -1          OUTPUT ORDER OF SCATTERING (DFLT=-1)
0NBUF   =    60          BUFFER SPACE (DFLT=60KB(IBM), 10KW(OTHER))
 INTYPE =     0          0/1/2 VARFLM/DOTIII/RTFLUX INPUT (DFLT=2)
 IOTYPE =     1          0/1/2 VARFLM/DOTIII/RTFLUX OUTPUT
 SPARE  =     0
 SPARE  =     0
 IGM    =     3          NUMBER OF ENERGY GROUPS
 IM     =    10          NUMBER OF I-INTERVALS
 JM     =     5          NUMBER OF J-INTERVALS
 ISCAI  =    -1          INPUT ORDER OF SCATTERING (DFLT=-1)
 MMI    =    -1          INPUT NUMBER OF DIRECTIONS (DFLT=-1)
0IGMI   =     0          NO. OF GROUPS INPUT (0 IMPLIES USE IGM)
 NEUT   =     3          NO. OF NEUTRON GROUPS
 ISM    =     1          NUMBER OF I SETS
 ITER   =     0          ITERATION NUMBER
 SPARE  =     0

0 2 * *
0EV     = 0.0            EIGENVALUE (KEFF FOR K-CALCULATIONS)
 DEVDK  = 0.0            EIGENVALUE SLOPE (FOR SEARCHES)
 EKEFF  = 0.0            KEFF (FOR SEARCHES)
 POWER  = 0.0            POWER
 XNORM  = 0.0            NORMALIZATION FACTOR (IGNORED IF 0)
```

Fig. 4.1. (Cont.)

```
OFINAL VALUES
0IMA    =    10          NUMBER OF I-INTERVALS
 IGMI   =     3          NO. OF GROUPS INPUT (DFLT=IGM)
 MMA    =    16          OUTPUT NUMBER OF DIRECTIONS
 MMI    =    16          INPUT NUMBER OF DIRECTIONS
 LM     =    10          NUMBER OF OUTPUT  FLUX MOMENTS
 LMI    =    10          NUMBER OF INPUT  FLUX MOMENTS
 IMSJM  =    50          INPUT MESH SIZE
 NBNDY  =   240          NUMBER OF OUTPUT BOUNDARY FLUXES
 NBNDI  =   240          NUMBER OF INPUT  BOUNDARY FLUXES
 NOM    =   450          LENGTH OF DOTIII OUTPUT ARRAY BEYOND 0TH
 NOMI   =   450          LENGTH OF DOTIII INPUT  ARRAY BEYOND 0TH

0    1151 WORDS MEMORY REQUIRED VS. AVAILABLE  126976
0    0T
0GROUP   =     1
0I-BOUNDARY FLUXES
```

0 M=	J= 1	J= 2	J= 3	J= 4	J= 5
1	0.0	0.0	0.0	0.0	0.0
2	1.43452E-05	6.88562E-06	3.72622E-06	5.34792E-06	1.49342E-07
3	7.34987E-03	6.02847E-03	4.91619E-03	3.91075E-03	1.16147E-03
4	0.0	0.0	0.0	0.0	0.0
5	2.71706E-03	1.44027E-03	2.84941E-04	3.23672E-05	5.94010E-06
6	2.67941E-05	4.61271E-05	1.62057E-05	8.06660E-06	4.38561E-06
7	6.81107E-03	5.61351E-03	8.22296E-03	8.17223E-03	3.07065E-03
8	5.75855E-03	7.13589E-03	7.40020E-03	5.94992E-03	2.47123E-03
9	0.0	0.0	0.0	0.0	0.0
10	2.63109E-07	3.52462E-06	3.39526E-06	1.40742E-05	1.18603E-05
11	3.76697E-05	1.29131E-02	5.48445E-02	9.60887E-02	8.58607E-02
12	0.0	0.0	0.0	0.0	0.0
13	1.95186E-05	7.93188E-05	2.65967E-04	1.75568E-04	3.97216E-05
14	4.60035E-06	1.69246E-05	2.75811E-05	3.75242E-05	3.48183E-06
15	5.91978E-05	1.59556E-03	1.03171E-02	2.45179E-02	1.62463E-02
16	6.43759E-05	5.77956E-03	2.61726E-02	2.03707E-02	1.44099E-02

0J-BOUNDARY FLUXES

0 M=	I= 1	I= 2	I= 3	I= 4	I= 5	I=
1	0.0	0.0	0.0	0.0	0.0	0.0
2	2.72702E-05	6.30102E-05	2.17421E-04	3.70463E-04	1.72564E-04	4.64575
3	6.76711E-06	2.40170E-05	5.28627E-05	1.11935E-04	1.82749E-04	2.50974
4	0.0	0.0	0.0	0.0	0.0	0.0
5	4.48093E-03	8.81993E-03	1.31676E-02	1.38670E-02	1.38126E-02	1.58441
6	7.94244E-05	2.65774E-04	9.03811E-04	3.57659E-03	9.06621E-03	1.29975
7	1.12528E-05	2.88813E-05	5.96236E-05	1.13940E-04	1.74594E-04	2.33645
8	9.64852E-06	2.49656E-05	4.08988E-05	5.75022E-05	6.92861E-05	8.34208
9	0.0	0.0	0.0	0.0	0.0	0.0
10	1.79695E-05	3.44707E-05	9.22805E-05	3.11216E-04	1.08996E-03	8.24779
11	3.32129E-06	1.02127E-05	2.42146E-05	7.32394E-05	1.95525E-04	7.58746
12	0.0	0.0	0.0	0.0	0.0	0.0
13	3.28171E-05	7.68654E-05	1.71830E-04	1.81074E-03	3.81943E-03	4.67112
14	1.72648E-05	6.80716E-06	1.18692E-04	7.04790E-05	6.47344E-03	1.31906
15	4.84064E-07	3.43850E-06	1.44269E-05	4.92749E-05	3.74894E-04	1.33542
16	2.55891E-07	2.87683E-06	8.02590E-06	2.47876E-05	1.08285E-04	2.12815

Fig. 4.1. (cont.)

```
0   M=      I=  9         I= 10
     1  0.0            0.0
     2  1.00285E-02    3.02962E-03
     3  1.29359E-02    1.12887E-02
     4  0.0            0.0
     5  8.02823E-03    1.01951E-03
     6  1.47215E-02    2.61670E-03
     7  9.81073E-03    9.99600E-03
     8  2.48843E-03    5.00195E-03
     9  0.0            0.0
    10  2.59098E-02    1.22831E-02
    11  9.07091E-02    8.69936E-02
    12  0.0            0.0
    13  3.47936E-03    1.47753E-03
    14  8.64138E-03    4.23882E-03
    15  2.85217E-03    9.52907E-03
    16  2.70870E-03    5.88802E-03
0GROUP   =      2
0I-BOUNDARY FLUXES
0   M=      J=  1         J=  2         J=  3         J=  4         J=  5
     1  0.0            0.0            0.0            0.0            0.0
     2  1.39102E-04    9.70713E-05    5.96749E-05    6.80444E-05    2.88709E-06
     3  5.64833E-03    4.96435E-03    4.54599E-03    3.78744E-03    1.15903E-03
     4  0.0            0.0            0.0            0.0            0.0
     5  3.39746E-03    2.14851E-03    5.97205E-04    1.20007E-04    3.33583E-05
     6  1.82331E-04    3.15633E-04    1.24254E-04    6.95911E-05    3.99101E-05
     7  4.10970E-03    3.52047E-03    5.09331E-03    6.09526E-03    2.45452E-03
     8  4.91681E-03    6.38171E-03    7.27557E-03    6.52250E-03    2.81978E-03
     9  0.0            0.0            0.0            0.0            0.0
    10  1.82538E-06    2.42120E-05    2.23280E-05    2.55343E-05    1.06781E-04
    11  2.85021E-05    5.10129E-04    3.00886E-03    7.83946E-03    8.66774E-03
    12  0.0            0.0            0.0            0.0            0.0
    13  6.29313E-05    2.11894E-04    5.32516E-04    3.83849E-04    1.29619E-04
    14  2.12164E-05    8.41362E-05    1.24178E-04    1.43478E-04    3.55675E-05
    15  7.62154E-05    9.72652E-04    4.39582E-03    8.33087E-03    5.82258E-03
    16  7.90503E-05    2.99975E-03    9.18279E-03    8.10320E-03    5.61294E-03
0J-BOUNDARY FLUXES
0   M=      I=  1         I=  2         I=  3         I=  4         I=  5         I=
     1  0.0            0.0            0.0            0.0            0.0            0.0
     2  2.07845E-04    4.29132E-04    1.25560E-03    1.61155E-03    4.71111E-04    1.04190
     3  4.38625E-05    1.47033E-04    3.18421E-04    6.27693E-04    9.64254E-04    1.28154
     4  0.0            0.0            0.0            0.0            0.0            0.0
     5  4.53785E-03    7.29259E-03    1.04856E-02    1.12161E-02    1.11021E-02    1.33382
     6  4.56674E-04    1.36077E-03    1.81088E-03    3.22432E-03    7.15413E-03    1.04954
     7  5.60218E-05    1.42327E-04    2.91008E-04    5.14172E-04    6.92570E-04    8.82336
     8  3.04749E-05    7.68892E-05    1.25964E-04    1.90279E-04    2.38439E-04    3.15996
     9  0.0            0.0            0.0            0.0            0.0            0.0
    10  8.36543E-05    2.26035E-04    4.50606E-04    1.35837E-03    2.76949E-03    3.20616
    11  3.86924E-05    1.27807E-04    2.75674E-04    7.39038E-04    1.46733E-03    2.40986
    12  0.0            0.0            0.0            0.0            0.0            0.0
    13  1.38714E-04    3.26487E-04    7.43280E-04    2.94074E-03    4.65857E-03    5.08127
    14  9.12655E-05    1.28728E-04    3.88179E-04    1.17603E-03    6.29907E-03    8.82735
    15  9.79831E-06    6.80481E-05    1.73280E-04    5.18666E-04    1.24527E-03    2.32078
    16  3.12926E-06    3.42142E-05    9.53869E-05    2.76950E-04    7.51590E-04    1.17926
```

Fig. 4.1. (Cont.)

0 M=	I= 9	I= 10
1	0.0	0.0
2	2.00400E-02	4.39140E-03
3	1.10541E-02	8.70653E-03
4	0.0	0.0
5	7.16840E-03	1.47848E-03
6	1.25434E-02	2.65246E-03
7	9.26367E-03	6.64536E-03
8	3.68065E-03	5.35728E-03
9	0.0	0.0
10	7.20767E-03	3.71511E-03
11	2.05056E-03	5.59365E-03
12	0.0	0.0
13	3.60226E-03	1.54544E-03
14	4.77364E-03	2.55699E-03
15	1.44732E-03	3.74172E-03
16	2.77446E-03	2.76781E-03

0GROUP = 3

0I-BOUNDARY FLUXES

0 M=	J= 1	J= 2	J= 3	J= 4	J= 5
1	0.0	0.0	0.0	0.0	0.0
2	1.43885E-04	1.05638E-04	6.60698E-05	7.57713E-05	3.84816E-06
3	2.18791E-03	2.15440E-03	2.47387E-03	1.78731E-03	5.25098E-04
4	0.0	0.0	0.0	0.0	0.0
5	1.66749E-03	1.13483E-03	3.73807E-04	9.29210E-05	3.09786E-05
6	1.85622E-04	2.53034E-04	1.13231E-04	6.69128E-05	2.79662E-05
7	1.50280E-03	1.54913E-03	2.74250E-03	2.52927E-03	1.00101E-03
8	2.15879E-03	2.93559E-03	3.46787E-03	3.16274E-03	1.37934E-03
9	0.0	0.0	0.0	0.0	0.0
10	2.19607E-06	2.90631E-05	2.63226E-05	2.64206E-05	8.68418E-05
11	3.08555E-05	4.03960E-04	1.72125E-03	3.09978E-03	2.48800E-03
12	0.0	0.0	0.0	0.0	0.0
13	3.40151E-05	1.13100E-04	2.59483E-04	1.84599E-04	7.31989E-05
14	2.01887E-05	7.48505E-05	1.02549E-04	1.06594E-04	3.12542E-05
15	6.35706E-05	5.80446E-04	2.33120E-03	3.01874E-03	2.03437E-03
16	7.18740E-05	1.90769E-03	3.89529E-03	3.60094E-03	2.24124E-03

0J-BOUNDARY FLUXES

0 M=	I= 1	I= 2	I= 3	I= 4	I= 5	I=
1	0.0	0.0	0.0	0.0	0.0	0.0
2	1.99540E-04	4.03490E-04	1.08819E-03	1.24069E-03	2.46275E-04	3.56280
3	4.95506E-05	1.61536E-04	3.43550E-04	6.45470E-04	9.55719E-04	1.21979
4	0.0	0.0	0.0	0.0	0.0	0.0
5	1.94933E-03	2.67088E-03	3.72277E-03	3.78443E-03	3.41890E-03	4.00094
6	3.83508E-04	1.08677E-03	1.11721E-03	1.27151E-03	2.61140E-03	3.30585
7	6.47673E-05	1.48543E-04	2.89381E-04	4.87279E-04	6.08210E-04	7.38369
8	4.69368E-05	1.10257E-04	1.66886E-04	2.30812E-04	2.63839E-04	3.29353
9	0.0	0.0	0.0	0.0	0.0	0.0
10	7.33615E-05	2.10937E-04	4.14148E-04	1.08272E-03	1.80349E-03	1.25378
11	4.25288E-05	1.43976E-04	3.13988E-04	7.56491E-04	1.31034E-03	1.77444
12	0.0	0.0	0.0	0.0	0.0	0.0
13	1.03854E-04	2.69122E-04	5.72727E-04	1.56364E-03	2.42790E-03	2.47946
14	8.24877E-05	1.36574E-04	3.82580E-04	8.67989E-04	2.14919E-03	3.18739
15	1.23753E-05	7.90888E-05	2.05744E-04	5.20435E-04	9.97589E-04	1.47375
16	4.21128E-06	4.27151E-05	1.19626E-04	3.38891E-04	7.61514E-04	1.04513

Fig. 4.1. (Cont.)

```
0    M=      I=  9         I= 10
     1  0.0           0.0
     2  4.08693E-03  1.77468E-03
     3  1.95673E-03  2.57234E-03
     4  0.0           0.0
     5  2.12643E-03  8.96250E-04
     6  3.10350E-03  1.29384E-03
     7  2.67880E-03  2.07940E-03
     8  1.32746E-03  1.85236E-03
     9  0.0           0.0
    10  2.44855E-03  1.35869E-03
    11  7.67169E-04  1.69617E-03
    12  0.0           0.0
    13  1.61760E-03  6.90681E-04
    14  1.72132E-03  9.82930E-04
    15  6.86930E-04  1.30563E-03
    16  1.76129E-03  1.17341E-03
0* * * TOTAL PROBLEM CPU =  0 HRS,  0.0028 MIN. * * * CPU INCREMENT =  0 HRS,  0
```

Fig. 4.2. Illustration of RTFLUM Problem Input.

```
=RTFLUM
VARFLM TO DOTIII, EDIT BOUNDARY FLUXES
1$$  2  3  10    A7  0  1    E T
T

DOTIII TO VARFLM, EDIT BOUNDARY FLUXES
1$$  3  2  10  6  3    60  1  0    2S    3  10  5    E T
T

VARFLM TO RTFLUX
1$$  2  4    A4  6  3    A7  0  2    E T
T

RTFLUX TO VARFLM, EDIT SCALAR FLUXES
1$$  4  2  1  6  3    E T
T

RTFLUX TO DOTIII, EDIT SCALAR FLUXES
1$$  4  2  1  6  3    A8  1    E T
T

DOTIII TO RTFLUX, EDIT SCALAR FLUXES
1$$  3  4  1  6  3    60  1  2    2S    3  10  5    E  T
T
=END
```

Section 5. Programmer Information for DOT Subsystem

W. A. Rhoades

5.1. Inter-machine Adaptability

The DOT subsystem codes are intended to be easily adaptable to any type of sophisticated computer, and yet to take advantage of certain localized structural features which may be machine-dependent.

In general, the guidelines of ANS-STD.3-1971[1] are followed. This requires general adherence to a simple, standard FORTRAN language except where deviations provide important improvements in capability and can be documented. In addition, machine-dependent features have been kept localized and have been kept away from the subroutines which perform actual computation.

The recommended procedures of the Committee on Computer Code Coordination (CCCC)[2,3] have been followed where practical. Compliance with these standards has been incomplete where cost was prohibitive, the nature of the codes did not allow it, or the expected level of performance could not be obtained in that fashion.

Where minor machine-dependent features are required, alternate features are enclosed in pairs of 3-character "language flags." The alternative statements remain in the source program, with inappropriate sections transformed into comment statements. If the code is set up for IBM operation, for example, it might contain:

```
CIB
        ENTRY IBCDC(H,E,L,P)
CIB
CDC
C       ENTRY IBCDC
CDC
```

The corresponding CDC configuration would be:

```
CIB
C       ENTRY IBCDC(H,E,L,P)
CIB
CDC
        ENTRY IBCDC
CDC
```

All versions of a given procedure are thus available for inspection by all users. The selection of options is made by a computer program at distribution time. A listing of machine-dependent sections can also be prepared in that process.

The language flags achieve the major objective of having a single, unified source for each program which is maintained for all users. With these provisions, the basic FORTRAN programs are operable on IBM, CDC, UNIVAC, CRAY, Amdahl, and other computers, except that system-dependent service routines as specified by CCCC must be provided.

Optional packages and procedures which provide enhanced convenience are also available. The optional run-time storage allocation requires system-dependent routines. Instructions for installing these options are distributed with the source programs. All known users at this time use all of the optional enhancements available to them. Even so, the basic operation with FORTRAN routines plus the CCCC package remains available if needed.

5.2. Service Subroutines

Certain standard service subroutines are specified by the CCCC for use in reactor physics codes. The service routines used in these codes include:

TIMER	Provides timing and job identification data
REED, RITE	Provide sequential access to data files
DRED, DRIT	Provide random access to data files
DOPC	Provides initiating, closing, and certain repositioning of data files
CRED, CRIT	Provide block transfer of data between fast and slow memory.

Many of these subroutines also call other subroutines. Many are entirely system dependent. Each configuration to be distributed contains an appropriate set of service subroutines, insofar as possible. The realities of computing environments may require local modification or substitution. The specifications given in Reference 3, together with extensive in-stream comments, provide guides for such modification.

5.3. References

1. ANS Standard "Recommended Programming Practices to Facilitate the Interchange of Digital Computer Programs," prepared by Subcommittee 10, ANS Standards Committee (April 1971).

2. B. M. Carmichael, "Standard Interface Files and Procedures for Reactor Physcs Codes, Version III," LA-5486-MS (February 1974).

3. R. Douglas O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes Version IV," LA-6941-MS (September 1977).

Section 6. Abstracts of Modules Previously Published

An Updated Version of the DOT 4 One- and Two-Dimensional Neutron/Photon Transport Code

Published as ORNL-5851 (1982)

W. A. Rhoades
R. L. Childs*

*Computer Sciences Division.

1.1. Program: DOT 4

1.2. Problem Solved: DOT 4 determines the flux or fluence of particles throughout a one- or two-dimensional geometric system due to sources either generated as a result of particle interaction with the medium or incident upon the system from extraneous sources. The principal application is to the deep-penetration transport of neutrons and photons. Criticality (k-type and search) problems can also be solved. Numerous printed edits of the results are available, and results can be transferred to output files for subsequent analysis.

1.3. Method of Solution: The Boltzmann transport equation is solved, using either the method of discrete ordinates or the diffusion theory approximation. In the discrete ordinates method, the primary mode of operation, balance equations are solved for the flow of particles moving in a set of discrete directions in each cell of a space mesh, and in each group of a multigroup energy structure. Iterations are performed until all implicitness in the coupling of cells, directions, groups, and source regeneration has been resolved. Several methods are available to accelerate convergence. Anisotropic cross sections can be expressed in a Legendre expansion of arbitrary order. Output data sets can be used to provide an accurate restart of a previous problem or to deliver information to other codes.

Several techniques are available to remove the effects of negative fluxes caused by the finite difference approximation, and of negative scattering sources due to truncation of the cross-section expansion. The space mesh can be described such that the number of first-dimensional (i)

intervals varies with the second dimension (j). The number of discrete directions can vary across the space mesh and with energy. Direction sets can be biased, with discrete directions concentrated such as to give fine detail to streaming phenomena.

1.4. Related Material:

Data Files

Microscopic cross-section input file

Independent source file (optional)

Flux guess input file (optional)

Flux result output files (optional)

Total source output file (optional)

Related Programs

DOS - a driver program which coordinates problem execution (IBM only)

BNDRYS - selects boundary fluxes for subsequent use as internal boundary sources

GIP - prepares cross-section input from card or tape input

GRTUNCL - prepares first-collision source due to a point source in RZ geometry (on or off axis)

RTFLUM - edits flux files and converts between various file formats

1.5. Restrictions: External force fields or non linear effects cannot be treated. Flexible dimensioning is used throughout, so that no restrictions

are imposed on individual problem parameters. Certain options, especially diffusion theory, are not compatible with variable mesh and quadrature problems.

1.6. Computers: DOT 4 is designed to be applicable to most sophisticated computers which support direct (random) access disk storage or the equivalent. It has special provisions for efficient use of a large, slow memory from which data are moved to fast memory in strings. Certain directly-addressed arrays of low usage can also be kept in slow memory, and slow memory is also used as a buffer between fast memory and certain disk files. Features which may be troublesome on new machines are restricted to replaceable packages. Proper operation has been demonstrated on many types of IBM, CDC, UNIVAC, and CRAY computers.

1.7. Running Time: Running time is roughly proportional to:

Flux work units (FWU) = number of space mesh cells x number of directions x number of energy groups x number of iterations per group.

Depending on the options chosen, a rate of one million FWU per minute on the IBM 370/3033 is typical. Thus, a very large problem with 5,000 space cells, 48 directions, 50 energy groups, 10 iterations per group, and P_3 scattering would require roughly 1 to 1.5 hours of CPU time. Execution on CDC or CRAY computers is usually faster.

1.8. Programming Languages: The standard FORTRAN described in ANS STD.3-1971 is used. The program can be operated with 100% FORTRAN, except for the subroutines necessary to implement I/O. On CRAY, CDC, and UNIVAC systems, these subroutines are obtained from the system library at load time.

On IBM systems, they are supplied with the code. On all systems, optional use of non-FORTRAN routines is provided in order to enhance convenience and speed.

1.9. Operating System: No special requirements are made of the operating system. IBM or UNIVAC overlay, CDC overlay, or CDC segmentation facilities may be used if available. The system must support direct (random) access data files.

1.10. Machine Requirements: Memory must be approximately 50,000 words for small problems. The requirement expands with problem size. External data storage must be provided for 9 scratch files, of which 5 must be direct (random) access. User-supplied input and output data files must be supplied on sequential-access devices, i.e., tapes or the equivalent.

1.11. Authors: Many persons have contributed to DOT development since 1965. Major contributors to this version are:

W. A. Rhoades

R. L. Childs

Major contributors to previous versions include:

W. W. Engle, Jr.

M. L. Gritzner

F. R. Mynatt

R. J. Rodgers

D. B. Simpson

E. T. Tomlinson

Questions concerning this code should be referred to the Radiation Shielding Information Center (RSIC), Oak Ridge National Laboratory, P. O. Box X, Oak Ridge, Tennessee 37830.

1.12. References:

1. F. R. Mynatt, F. J. Muckenthaler, and P. N. Stevens, "Development of a Two-Dimensional Discrete Ordinates Transport Theory for Radiation Shielding," CTC-INF-952 (August 1969).

2. K. D. Lathrop and F. W. Brinkley, "TWOTRAN-II: An Interfaced, Exportable Version of the TWOTRAN Code for Two-Dimensional Transport," LA-4848-MS (July 1973).

3. W. A. Rhoades and R. L. Childs, "An Updated Version of the DOT 4 One- and Two-Dimensional Neutron Photon Transport Code," ORNL-5851 (1982).

1.13. Material Available:

User's Manual

Source File in Update Form

Job Control Language for Unloading Source File and Installing Code

Sample Problem Solutions

1.14. Abstract Author: W. A. Rhoades, ORNL.

A User's Manual for ANISN, A One-Dimensional Discrete Ordinates Transport Code with Anisotropic Scattering

Published as K-1693 (1967)

W. W. Engle, Jr.

1.1. Name: ANISN

Title: A One-Dimensional Discrete Ordinates Transport Code with Anisotropic Scattering

Auxiliary Programs: GIP - prepares cross section input

DOGS - provides graphical output

1.2. Contributors: W. W. Engle, Jr.

Oak Ridge National Laboratory

P.O. Box X

Oak Ridge, TN 37830

1.3. Language: FORTRAN IV

1.4. Nature of Problem: ANISN determines the flux of particles throughout a one-dimensional geometric system due either to sources generated by particle interactions with the medium or to sources which are independent of the system.

1.5. Method of Solution: The Boltzman transport equation is solved using the method of discrete ordinates. Space dependent rebalance using an automatic coarse mesh calculation is used to accelerate the convergence of the flux iterations.

1.6. Restrictions: External force fields and non-linear effects cannot be treated. Variable dimensioning is used in all routines so that there are no restrictions on individual problem parameters.

1.7. Typical Running Time: Running time is proportional to the number of space, angle and energy dependent fluxes which are calculated and can vary from less than a second to more than an hour on the IBM 3033 System.

1.8. Hardware Requirements: Memory must be approximately 50,000 words for a reasonable problem, expanding or contracting with problem size. Sequential scratch storage is required and input and output files must be supplied on sequential devices.

1.9. Software Requirements: The program is operable with 100% FORTRAN but the standard ORNL version uses the machine language routine ALOCAT to achieve run-time allocation of memory.

1.10. References:

1. W. W. Engle, Jr., "A User's Manual for ANISN, A One-Dimensional Discrete Ordinates Transport Code with Anisotropic Scattering," K-1693 (March 1967).

DOPES: Discrete Ordinates Perturbation System

E. T. Tomlinson*
R. L. Childs**
R. A. Lillie

*TVA, Chattanooga, TN.

**Computer Sciences Division.

1.1. Program: DOPES: Discrete Ordinates Perturbation System

1.2. Problem Solved: The Discrete Ordinates Perturbation System can be used to perform perturbation-theory calculations using either diffusion or transport theory, and first order or exact method. It can also be used in other applications where the inner product of two fluxes is necessary, i.e., sensitivity calculations.

1.3. Method of Solution: The DGRAD module calculates cell averaged partial currents or partial flux gradients for both forward and adjoint fluxes based on the geometry, cross sections and diffusion theory flux files. The VIP module calculates zone-dependent ϕ,ϕ^* tables for use in perturbation analyses for either regular and uncollided flux files using either transport-theory fluxes (with flux moments included) or diffusion-theory fluxes from DGRAD. The TPERT module can be used to calculate perturbations to systems using transport or diffusion perturbation theory. Either first order or exact perturbation theory can be used to obtain k_{eff}, dose rate, etc. perturbations. Each component of the perturbation is edited separately, i.e., leakage, fission, removal, etc.

1.4. Related Codes and Materials: The GIP, DOT, and RTFLUM programs may be used in preparing input files for DOPES.

1.5. Restrictions: The variable-mesh features of DOT are not presently available in DOPES.

1.6. Computers: Although the DOPES codes should be adaptable to many types of computers, only IBM 360/370 operation has been demonstrated.

1.7. Running Time: DOPES problems require from 1 second to a few minutes on conventional computers, depending upon the problems solved.

1.8. Programming Languages: DOPES members are programmed in standard FORTRAN, insofar as possible. In several instances, optional IBM assembler-language routines which enhance program capability or convenience are provided.

1.9. Operating System: The DOPES modules presently operates under MVS/JES2.

1.10. Machine Requirements: Useful problems can be solved in 270 K-bytes of memory on a machine having at least the capability of the IBM 360/75.

1.11. Authors or major contributors:

E. T. Tomlinson, Union Carbide Nuclear Division, Oak Ridge, TN

R. L. Childs, Union Carbide Nuclear Division, Oak Ridge, TN

R. A. Lillie, ORNL, Oak Ridge, TN

1.12. References:

1. E. T. Tomlinson, R. L. Childs, and R. A. Lillie, "DOS Perturbation Modules DGRAD/VIP/TPERT", ORNL/CSD/TM-116 (April, 1980).

1.13. Material Available:

Source Program

User's Manual (Reference 1)

1.14. Abstract Prepared By: E. T. Tomlinson

DOGS - A Collection of Graphics for Support of Discrete Ordinates Codes

Published as ORNL/TM-7188 (1980)

D. T. Ingersoll
C. O. Slater

1.1. Name: DOGS

Title: A collection of routines for the graphical display of calculated data generated by discrete-ordinates codes.

Individual Routines: EGAD - Plot 2-dimensional geometries

ISOPLOT4 - Plot 2D flux contours

FORM - Plot 2D surface maps

ACTUAL - Calculate activities

TOOTH - Plot 1D contributon surfaces

ASPECT - Plot energy/space distributions

Auxiliary Routines: DOGS routines read flux files written by:

ANISN - 1 D transport code

XSDRNPM - 1D transport code

DOT - 2D transport code

1.2. Contributors: D. T. Ingersoll and C. O. Slater

Oak Ridge National Laboratory

P.O. Box X

Oak Ridge, TN 37830

1.3. Language: FORTRAN IV and Assembler

1.4. Nature of Problem: To produce graphical display of flux data calculated by common discrete-ordinates radiation transport codes.

1.5. Method of Solution: DOGS uses the DISSPLA proprietary graphics software package to provide flexible plot construction and manipulation.

1.6. Restrictions: All of the DOGS routines use flexible dimensioning, and therefore, no restrictions are placed on problem size. All of the routines except ACTUAL require the availability of DISSPLA software.

1.7. Typical Running Time: The execution time depends on the number of plots being generated. The time required for each plot is on the order of a few seconds.

1.8. Hardware Requirements: The DOGS routines are operational on IBM 360/91, 370/195, and 3033 systems. Most of the routines require a sequential scratch storage device and TOOTH requires a random access device. All routines except ACTUAL require a plotting device.

1.9. Software Requirements: All routines except ACTUAL requires the DISSPLA graphics package sold by Integrated Systems Software Corp. Also, all routines use the assembly language subroutine ALOCAT to achieve run-time allocation of memory space. This can be avoided by compiling a fixed dimension for the main container array. TOOTH additionally requires the assembly language routines DEFILE and CLOSDA to manipulate random access storage device.

1.10. References:

1. D. T. Ingersoll and C. O. Slater, "DOGS - A Collection of Graphics for Support of Discrete Ordinates Codes," ORNL/TM-7188 (March 1980).

ORNL/TM-8362
Distribution Category UC-79d

INTERNAL DISTRIBUTION

1-2. L. S. Abbott
3. R. G. Alsmiller, Jr.
4. J. M. Barnes
5. D. E. Bartine
6. B. L. Broadhead
7. D. G. Cacuci
8. R. L. Childs
9. H. L. Dodds
10-14. M. B. Emmett
15. W. W. Engle, Jr.
16. G. F. Flanagan
17. W. E. Ford, III
18. T. A. Gabriel
19. U. Gat
20. O. W. Hermann
21. D. T. Ingersoll
22. J. O. Johnson
23. H. E. Knee
24. J. R. Knight
25. R. A. Lillie
26. R. E. Maerker
27. F. C. Maienschein
28. J. H. Marable
29. B. F. Maskewitz
30. D. L. Moses
31. F. C. Mynatt
32. J. V. Pace, III
33. L. M. Petrie
34. R. T. Primm, III
35. J. A. Renier
36-48. W. A. Rhoades
49. R. W. Roussin
50. J. C. Ryman
51. R. T. Santoro
52. D. L. Selby
53. C. O. Slater
54. J. S. Tang
55. D. R. Vondy
56. C. R. Weisbin
57. R. M. Westfall
58. J. E. White
59. J. R. White
60. G. E. Whitesides
61. L. R. Williams
62. M. L. Williams
63. B. A. Worley
64. A. Zucker
65. P. W. Dickson, Jr. (Consultant)
66. H. J. C. Kouts (Consultant)
67. W. B. Lowenstein (Consultant)
68. R. Wilson (Consultant)
69-70. Central Research Library
71. Y-12 Document Ref. Section
72-73. Laboratory Records Dept.
74. Laboratory Records ORNL, RC
75. ORNL Patent Office
76-110. EPIC

EXTERNAL DISTRIBUTION

111. Assistant Manager for Energy Research & Development
DOE, Oak Ridge, TN 37830

112. Division of Reactor Research & Development
Department of Energy
Washington, DC 20545

113. Richard Archibald
General Atomic Company
P.O. Box 81608
San Diego, CA 92138

114. Don Mathews
General Atomic Company
P. O. Box 81608
San Diego, CA 92138

115. Morgan G. Libby
General Electric Company
Mail Code S-34
P.O. Box 510
Sunnyvale, CA 94086

116. E. T. Tomlinson
Tennessee Valley Authority
409 Krystal Building
Chattanooga, TN 37401

117. Mr. Eugene Specht
Atomics International
P.O. Box 309
Canoga Park, CA 91304

118. Mr. Leo B. Levitt
Atomics International
P.O. Box 309
Canoga Park, CA 91304

119. Dr. R. D. O'Dell
Los Alamos Scientific Laboratory
P.O. Box 1663, Stop T-1
Los Alamos, NM 87544

120. Dr. Siegfried A. W. Gerstl
Los Alamos National Laboratory
Theoretical Div.
Box 1663 T-DOT MS 210
Los Alamos, NM 87545

121. Mr. Samuel L. Stewart
General Electric - ARSD
310 Deguigne Dr.
Sunnyvale, CA 94086

122. Mr. W. H. Harless
General Electric Co.
310 Deguigne Dr.
Sunnyvale, CA 94086

123. Lt. Col. James H. Lee, Jr.
Bldg. 413, Rm. 262
Aberdeen Street
Air Force Weapons Laboratory
Kirtland Air Force Base
Albuquerque, NM 87117

124. Mr. Philip B. Hemmig
Chief of Physics Section
RRT Division
U.S. Dept. of Energy
MS B107
Washington, DC 20585

125. Dr. J. W. Lewellen
Division of Reactor Research
and Development
U.S. Dept. of Energy
Washington, DC 20545

126. Dr. Sheldon Levin
Armed Forces Radiobiological
Research Laboratory
National Naval Medical Center
Bechesda, MD 20014

127. Dr. D. L. Auton
Defense Nuclear Agency
6801 Telegraph Rd.
Alexandria, VA 22310

128. Mr. C. J. Hamilton
General Atomic Co.
P.O. Box 81608
San Diego, CA 92138

129. Dr. Carl A. Rouse
General Atomic Co.
P.O. Box 81608
San Diego, CA 92138

130. Mr. R. K. Disney
Westinghouse Electric Co.
P.O. Box 158
Madison, PA 15663

131. Mr. G. N. Wrights
Westinghouse Electric Co.
P.O. Box 158
Madison, PA 15663

132. Mr. W. L. Bunch
Hanford Engineering
Development Laboratory
Westinghouse Hanford Co.
Richland, WA 99352

133. J. L. Fletcher
Knolls Atomic Power
Laboratory
P.O. Box 1072
Schenectady, NY 12301

134. O. W. Lazarus
Bldg. 129
Brookhaven National
Laboratory
Upton, NY 11973

135. W. L. Thompson
Group TD-6, MS 226
Los Alamos Scientific Laboratory
Box 1663
Los Alamos, NM 87545

136. D. J. Dudziack
Los Alamos Scientific Laboratory
Box 1663
Los Alamos, NM 87545

137. Leo Lesage
Argonne National Laboratory
Bldg. 316
9700 S. Cass Ave.
Argonne, IL 60439

138. H. M. Murphy, Jr. (DYM)
Air Force Weapons Laboratory
Kirtland Air Force Base
Albuquerque, NM 87117

139. Lt. Commander Robert Devine
Armed Forces Radiobiological
Research Laboratory
National Naval Medical Center
Bethesda, MD 20014

140-201. Given DNA Radiation Transport Distribution.

202-323. Given Distribution as shown in TID-4500, Distribution Category UC-79d, Breeder Reactor Physics (Base).