

NRTSC
NUCLEAR REACTOR TECHNOLOGY
AND SCIENTIFIC COMPUTATIONS

KEY WORDS:
COMPUTER CODE
REACTOR PHYSICS
THERMAL-HYDRAULICS

RETENTION PERIOD:
UNLIMITED

**VERIFICATION AND VALIDATION PLAN
FOR REACTOR ANALYSIS COMPUTER CODES (U)**

By

**H. Toffer
R. D. Crowe
K. N. Schwinkendorf
Westinghouse Hanford Company**

**R. E. Pevey
Scientific Computations Division
Savannah River Laboratory**

ISSUED: NOVEMBER 1989

**SRL SAVANNAH RIVER LABORATORY, AIKEN, SC 29808
Westinghouse Savannah River Company
Prepared for the U. S. Department of Energy under
Contract DE-AC09-80SR18035**

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

PROJECT: Code Certification

DOCUMENT: WSRC-RP-89-1249

TITLE: Verification and Validation Plan for Reactor
Analysis Computer Codes (U)


TASK:

APPROVALS



M. R. BUCKNER, MANAGER
SCIENTIFIC COMPUTATIONS

DATE: 11-15-89



C. E. APPERSON, MANAGER
REACTOR PHYSICS

DATE: 11-15-89

TABLE OF CONTENTS

1.0 SUMMARY	2
2.0 INTRODUCTION	3
3.0 OBJECTIVES OF PLAN	4
4.0 BASIC ASSUMPTIONS	5
5.0 LIST AND DESCRIPTION OF CODES COVERED	5
6.0 QA REQUIREMENTS	6
7.0 ACTION MATRIX	6
8.0 RESOURCE REQUIREMENTS	19
9.0 TIME SCHEDULE	19
10.0 CONCLUSIONS	19
11.0 REFERENCES	21
APPENDIX	A - 1

**VERIFICATION AND VALIDATION PLAN
FOR REACTOR ANALYSIS COMPUTER CODES (U)**

1.0 SUMMARY

A verification and validation (V&V) plan for reactor analysis computer codes used in Technical Specifications development and for other safety and production support calculations has been prepared. This plan fulfills the commitments by Westinghouse Savannah River Company (WSRC) to the Department of Energy Savannah River (DOE-SR) as identified in a letter to R.E. Tiller (Reference 1).

The plan stresses verification and validation by demonstrating successful application of the codes to predict reactor data, special measurements, and benchmarks. This is in compliance with the intent of the WSRC quality assurance requirements. Restructuring of software especially to achieve verification compliance is not recommended.

An action matrix, a time schedule, and a resource commitment table have been included in the plan. These items identify what is required to achieve verification and validation of the codes, the time table that this will be accomplished on, and the resources needed to support such an effort.

A list of computer codes covered by the verification and validation plan has been established. A description of each of the codes is provided. The action matrix identifies specific requirements that need to be met to achieve the verification and validation plan's objectives.

Large segments of information called for in the action plan have been generated in the past; however, the proper documentation needs to be identified, assembled, and properly updated. Some validation and benchmark calculations will have to be performed for the latest version of the computer codes.

The development of the plan is based in part on the experience and insights of Westinghouse Hanford Company in their successful efforts to establish computer code compliance to quality assurance requirements.

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

2.0 INTRODUCTION

As a consequence of Department of Energy (DOE) reviews of Savannah River Site (SRS) reactor operations, the DOE requested Westinghouse Savannah River Company (WSRC) in reference letter 2 to demonstrate that the process employed to develop parameters used in Technical Specifications is valid. This requirement calls for a key documented baseline and/or validation process for the computer codes essential to Technical Specification development. A verification and validation plan has been developed to respond to this request. The plan addresses reactor physics and thermal hydraulics codes, but the plan's structure is sufficiently general to be useful for a variety of types of applications in the future. Subsequent revisions of this document could list such codes or other reactor analysis tools adapted to solution of SRS reactor problems.

Some of the codes covered by this plan have been written as many as 20 years ago. At that time quality assurance requirements on software development, testing, and control were neither defined in present day context nor did they contain current rigor. Records maintained of code development activities, testing, and validation are difficult to recover. A documented configuration control record for most codes is incomplete. Despite these shortcomings in light of present day quality assurance requirements, the codes have been successfully used to predict the operating characteristics of the reactors, safety parameters, isotope production and certain key measurements for many years. The plan will establish the means and schedule to rectify the shortcomings to establish documented verification and validation of the codes. Emphasis in this effort will be in demonstrating the successful application of the codes and their ability to predict reactor conditions and test data and thereby bring the codes into conformance with quality assurance requirements.

The computer codes covered by this plan are those used directly in the computation of key Technical Specification parameters or employed to indirectly support Technical Specifications, charge design, or reactor operating limits.

The plan has been prepared in cooperation with Westinghouse Hanford Reactor Applications personnel acting as consultants to the

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

SRS Reactor Physics organization. The Westinghouse Hanford Company has gone through a similar activity within the last few years in support of the N Reactor Safety Upgrades Program, the N Reactor Flux Flattening Program, and the Alternate Missions Program. Their experience, insights, and outside perspective have been essential and invaluable to the development of the plan.

The successful implementation of the plan requires an understanding of what is meant by code verification and code validation and the relationship of these terms to the SRS quality assurance documentation.

Verification denotes the process which establishes that the theory is correct and has been properly coded and that the various code modules are functionally coupled to process information as required.

Validation is the process that establishes how well a computer code can reproduce observed or measured reality such as experimental data obtained in special facilities, controlled experiments in operating facilities, operational data or benchmarks. Benchmarking refers to the process of evaluating the performance of one computer code relative to another code or relative to an exact solution.

3.0 OBJECTIVE OF PLAN

The objective of the Verification and Validation Plan is to establish verification and validation of the computer codes and their application as spelled out in the letter to R.E. Tiller, August 11, 1989 (Reference 1).

The specific points in the reference include:

- o Identification of computer codes requiring verification and validation,
- o Development of benchmark packages, including standard problems, to ensure proper definition of the requirements for portability and limitations for the codes,

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

- o Documentation of an experimental basis, where practical, for the code calculations and linking the results to tests conducted, and
- o Ensuring that user documentation is complete and identified.

4.0 BASIC ASSUMPTIONS

For existing software, where the codes were developed outside the framework of the current Quality Assurance requirements (Reference 3), there exists an approach to achieve compliance with such requirements. The approach emphasizes demonstrating that the codes have been successfully used to predict operating conditions, special measurements and conformance to industry benchmarks in the past and thereby should not require coding reconfiguration.

It is anticipated that the documentation associated with codes developed a number of years ago can be used in its original or upgraded form to meet some of the documentation requirements. There is a large set of experimental data that have been used or could be used for code testing.

5.0 LIST AND DESCRIPTIONS OF CODES COVERED

A list of pertinent reactor physics, thermal hydraulics, and safety analysis codes has been assembled. The covered codes are:

AA3	FLOWTRAN-FI
FLOWTRAN-TF	FLOWZONE
GILDA	GLASS
GRIMHX	HMTABLE
ICG	JASON
LLAP	MARCO
PIPEFLOW	PLENUM
PORAD	RELAP5
SCALEUP	SHIELD
TRAC/MOD1	TRIMHX
WIGGLE	

Code descriptions for the above listed codes are contained in the Appendix.

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

6.0 QA REQUIREMENTS

It is the intent of this Verification and Validation Plan to parallel the QA requirements for code certification relating to Software Benchmark Testing as described in Section QAP IV-9 of Reference 3. Thus the work performed for this plan is directly applicable to tasks required for Code Certification under the present QAP manual. Completion of the Validation and Verification for each code represents a major step in Code Certification.

Sections IV-2, Certification of Existing Software and IV-9, Benchmark Testing, of the NRT&SC QA Procedures will be reviewed as part of this effort. This will facilitate incorporating the verification and validation effort into the overall code certification task.

7.0 ACTION MATRIX

An action matrix for the V&V plan has been developed and is shown in Figures 1 and 2. The matrix identifies the different types of information that can be assembled for each code to establish code verification and validation. The matrix serves as a useful tracking mechanism for monitoring the completion of activities. In the matrix, the names of the various codes are listed, as well as a series of action items. The activities are grouped by topics. The BASIC group deals with basic requirements for each code. The topics in the THEORY group pertain to establishing a verification of the codes. The items listed under EXPERIMENTS and BENCHMARKS are covered within the validation effort. The CONCLUSIONS category deals with the completion process for the verification and validation action matrix (V&VAM). Additional information on the V&VAM is contained in this section as detailed descriptions of what is needed to satisfy each topic.

Not all the boxes are relevant for each computer code. For example, an analytic solution may not exist for a particular code, or perhaps one particular computer code only reformats some data for input into another code and a theoretical description may not be

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

appropriate. Consequently, there are two ways for each box to be completed and checked off. Either by performing the tasks identified for each box or demonstrating that this specific box is not relevant for this particular code. In either case, this information is to be included in the documentation for the particular box.

The following sections give a detailed description of what is called for in the specific action items.

7.1 BASIC REQUIREMENTS

7.1.1 User Manuals in Place

Is adequate documentation in place for a person to use the code?

Adequate user documentation should exist so that a person who has technical familiarity and background in the appropriate field would be able to gain sufficient knowledge of the computer code's input parameters to execute relevant calculations on a controlled version. A knowledge of the specific computer's operating system would be assumed.

7.1.2 Configuration Control Plan

Has a configuration control plan been established for the particular code?

A code that is being validated should be under a configuration control plan. The version of the code should be frozen, users and theory manuals should be in place, and controlled approaches to performing the validations should be established.

7.1.3 Code Portability

Is the computer code portable from one operating system or computer to another one?

The computer environment is changed periodically. New operating

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

systems are implemented and more advanced computers are installed. The computer code should be written in a language and in a manner that lends itself to portability. Depending on the application of a particular code, portability may be dictated by its application.

7.2. Verification Of Theory

7.2.1 Appropriate Theory

Does an established, accepted theoretical basis exist for the code?

Computer codes that model a physical process, entity or condition require a theoretical or empirical description of the phenomenon being investigated. This description or model must be an appropriate representation of the actual physical situation for the computer code. The theory must be relevant to the problem being solved such that all pertinent phenomena and parameters are considered and the treatment of independent variables is correct. An exact mathematical description may be beyond existing computational capabilities and therefore approximations to the theory may have to be introduced. The effect of these approximations and the conditions for which they are valid must be evaluated before the computer code theory can be accepted as appropriate.

7.2.2 Theory Documented

Has the theory for the code been documented?

The basis for the theoretical treatment should come from established engineering theory and practices when possible. A wealth of documentation material to substantiate the theory such as textbooks, reports and journals are available. A theoretical development not found in the open literature could be internal documentation with proper technical review.

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

7.2.3 Coding Consistency

Has the theory been coded properly and is the software functioning as required?

The solutions or mathematical descriptions have to be translated into a computer code language. There must be a consistency between the coding and the calculational logic. Coded modules have to interact functionally to meet code operation objectives. For computer codes developed several years ago, this process, although performed, may not have been documented and consequently, the check of the coding consistency for these codes may be in the demonstrated ability of the code to predict physical observables.

7.2.4 Theory Verified Conceptually

Was the theory verified with physical concepts rather than experimental data?

The development of a theoretical description for a particular problem may involve combining several specific concepts. The interaction between the theoretical models and underlying approximations has to be identified and tested to insure that the concepts and application of the individual theoretical models combine to create a valid unified theoretical description.

7.2.5 Theory Verified by Experiment

Was the theory verified with experimental data?

Most theoretical descriptions of a problem can be tested and verified by experimental measurements. A theory that has demonstrated its validity by predicting or interpreting experimental results for one set of conditions can be applied to a new set of problems, provided the differences in conditions is understood.

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

7.2.6 Theory Documentation Adequate

Does a document exist that adequately explains the theory for the code?

Documentation is required to demonstrate the appropriateness of the theoretical development. Is there conclusive documentation to support the assertions made in answer to the previous questions? This documentation should be in the open literature or an internal document and have been subject to technical review.

7.3 Validation With Experiments

7.3.1 Tests in Experimental Facilities

Have tests been performed in special facilities that are applicable to code validation?

The accuracy of a computer code in replicating a physical situation can be established by comparing to data obtained from special test facilities. The first step in this process would be to identify relevant test data. The value of each individual test will depend on how closely it corresponds to the computer code's intended purpose, the range of variables measured during the test and the quality and accuracy of the data collected. Additional tests or measurements could be identified that would utilize experimental facilities to refine or determine a code's accuracy for a particular situation.

7.3.2 Tests in Operating Facilities

Have controlled tests been performed in operating facilities that can be used for code validation?

Another way of measuring the accuracy of a code is by comparison to measurements made during special experiments in operating facilities. While experimental facilities may allow measurement of conditions that exceed allowable conditions for an operating facility, the experimental facilities may not replicate prototypic conditions. Therefore, the value of special tests in operating facilities is

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

important to establish values of parameters in operating environments. Again, the value of these data depend on the accuracy and quality of the data.

7.3.3 Data from Operating Facilities

Are data available from operating facilities for code validation?

During normal operation, a large quantity of data is regularly collected. These data are a very useful source of information for validating the functions of a computer code. The data relevant for comparison with the computer code predictions need to be identified. This information may exist in stored data records of operating conditions or may have to be flagged for collection during normal operation after restart. The quality of older data has to be evaluated whereas the quality of data collected in the future can be controlled.

7.3.4 Test Data Documented

Are the data from any of the three sources documented?

The critical consideration in determining the value of existing test data for code validation is the quantity and quality of the documentation describing the conditions of the tests and controls used in collecting the data. The documentation requirements apply to all three sources of data before they can be used for validation.

7.3.5 Appropriate Data Quality

Is the quality and applicability of the data such that it can be used for validation?

The experimental or test data used for code validation should be collected under experimental or operating conditions subject to the

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

applicable quality assurance (QA) requirements (QAP III). The conditions imposed by the QA requirements insure that the test data are of appropriate quality. In the case of older test data, taken before present QA Standards were adopted and having incomplete QA records, the QA manual describes several methods to be used to qualify these test data, including use of corroborating data, confirmatory testing, analysis of controls used and technical review.

7.3.6 Validation Performed

Have validations been performed with any or all data sources?

Once the quality of the validation data has been established, code testing shall be conducted in accordance with the applicable technical procedures. The results of the validation computations shall be compared with the test data to provide a performance evaluation of the computer code for the specific application. This should include evaluating the ability of the software to model processes or phenomena over a specific range of variables. The comparison should attempt to demonstrate code applicability over a large range of input variables. Validation has to be performed with either configuration controlled software or in accordance with a specified configuration control plan. The results of the validation for critical applications must be reported and undergo technical review as described in QA procedures. Past validations have to be accepted on the details included in the documentation.

7.3.7 Validation Documentation Adequate

Has the validation been documented in an adequate manner?

Adequate documentation for validation requires complete records of the test data used for the validation, identification of any supporting software used including data files, codes, library routines and systems software and specification of the valid range of inputs and

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

outputs of the code. Applicable QA procedures exist for determining the completeness of the documentation, for example QAP IV-9. Past validation documents have to be judged on how well they meet the intent of present QA requirements.

7.4 Validation by Benchmarking

7.4.1 Benchmark Requirements Identified

Have the benchmark and specific requirements for the code in question been established?

The benchmark tests need to exercise the full range of the code's capabilities to demonstrate the computer code performs as intended. The tests should be as representative as possible of the actual conditions for which the computer code is intended. The range of inputs and anticipated outputs needs to be identified. The methods to be used to compare the benchmark data and calculated results should be specified.

7.4.2 Similar Code Comparison

Are similar approved codes available that could duplicate the desired calculations and thereby provide a benchmark?

This type of code benchmarking requires identifying another computer code that performs similar calculations. Both computer codes attempt to solve the same problem and the resulting calculations are compared.

7.4.3 Exact Solution Comparison

Are exact solutions available against which the computer code can be tested?

In many cases, the model description used in some computer codes can be simplified to allow for an analytical or exact solution. This can give an absolute measure of the code performance. Such

**VERIFICATION AND VALIDATION PLAN
FOR REACTOR ANALYSIS COMPUTER CODES (U)**

solutions can provide part of the benchmarking even if only a limited range of the inputs is exercised.

7.4.4 Industry Benchmark Comparison

Are there industry accepted benchmark problems available against which the computer code can be tested?

Several benchmarks problems have been identified by industry and are used to test new codes. Some of these benchmarks may be appropriate for testing of the codes on the list. In such a case, they would provide a significant test for the computer codes being validated.

7.4.5 Comparisons Documented

Have the comparisons to other codes, to exact solutions and/or industry accepted benchmarks been documented?

The goal of the benchmark tests is to evaluate the ability of the software to model the processes or phenomena over a specific range of variables. The results of the analysis must be documented to include identification of the sources of comparison data, the range of specific inputs, methods of evaluating the test results and description of how the testing was conducted.

7.4.6 Benchmark Documentation Adequate

Is the documentation of the benchmark calculations adequate to pass a technical review?

Adequate documentation for benchmarking requires complete records of the comparison test data used for the tests, identification of any supporting software including data files, codes, library routines and systems software and specification of the range of inputs and outputs of the code tested. Applicable QA procedures exist for determining the completeness of the software benchmark

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

testing, see QAP IV-9. Past benchmarking test documents have to be judged on how well they met the intent of present QA requirements.

7.5 Concluding Tasks

7.5.1 Verification Review

Has a code verification review been completed?

Every code should be passed through a technical review to establish if the verification steps completed in the theory section are sufficiently complete to declare that the code is considered verified. A technical review group would have to be established composed of SRS personnel with support from consultants and function under the accepted QA requirements. A report generated by this group would recommend acceptance of verification for a particular code.

7.5.2 Verification Completed

Has the computer code verification process been completed?

Following the technical review group recommendation, verification of a particular code is accepted by the management of the specific organization. The verified code under its configuration control plan will be assigned to the control of a specific code custodian. It will be his/her responsibility to ensure that code verification is maintained through subsequent configuration and computer system changes. A report by the appropriate manager will satisfy the Department of Energy Savannah River Office of the completion of this activity.

7.5.3 Standard Set of Test Problems

Has a standard set of test problems been established for validation and benchmarking purposes?

As an outcome of the validation effort a standard set of documented

VERIFICATION AND VALIDATION PLAN FOR REACTOR ANALYSIS COMPUTER CODES (U)

test problems should evolve that can be processed subsequent to any code modification to ensure consistency. The standard set of test problems can include comparisons to experimental data, other codes, exact solutions, and industry accepted benchmarks. The set should span the potential range of applications for the code and should test features and special data records of interest.

7.5.4 Validation/Benchmarking Review

Has a code validation/benchmarking review been completed?

Every code should be passed through a technical review to establish if the validation/benchmarking steps are sufficiently complete to declare that the code is considered validated and benchmarked. A technical review group would have to be established composed of SRS personnel with support from consultants and function under the accepted QA requirements. A report generated by this group would recommend acceptance of validation/benchmarking for a particular code.

7.5.5 Validation/Benchmarking Completed

Has the computer code validation/benchmarking process been completed?

Following the technical review group recommendation, validation and benchmarking of a particular code is accepted by the management of the specific organization. The validated and benchmarked code under its configuration control plan will be assigned to the control of a specific code custodian. It will be his/her responsibility to ensure that code validation and benchmarking are maintained through subsequent configuration and computer system changes. A report by the appropriate manager will satisfy the Department of Energy Savannah River Office of the completion of this activity.

**VERIFICATION AND VALIDATION PLAN
FOR REACTOR ANALYSIS COMPUTER CODES (U)**

8.0 RESOURCE REQUIREMENTS

The estimated manpower required to complete the verification and validation effort are given below:

	FY 1990	FY 1991
Physics	34 man-months	34 man-months
Thermal-hydraulics	14 man-months	14 man-months

TOTAL	48 man-months	48 man-months

9.0 TIME SCHEDULE

The time schedule for completion of the verification and validation efforts for all codes is given in Figure 3. Milestones 1, 2, and 3 are preliminary to the effort leading to the completion of milestones 4 through 7. Milestone 3 establishes what pertinent information is in existence and what needs to be done for each specific code to bring it in compliance with the verification and validation plan requirements.

10. CONCLUSIONS

A Verification and Validation Plan has been established that brings certain computer codes essential to Technical Specifications development and other codes needed for safety analysis and production predictions in compliance with verification and validation requirements. Completion of the items in the action matrix with the resources identified will assure that code verification and validation can be accomplished according to the established time schedule. The plan is sufficiently general that it can be applied to other computer codes in the area of reactor and criticality safety or to new reactor analysis codes adapted to SRS applications.

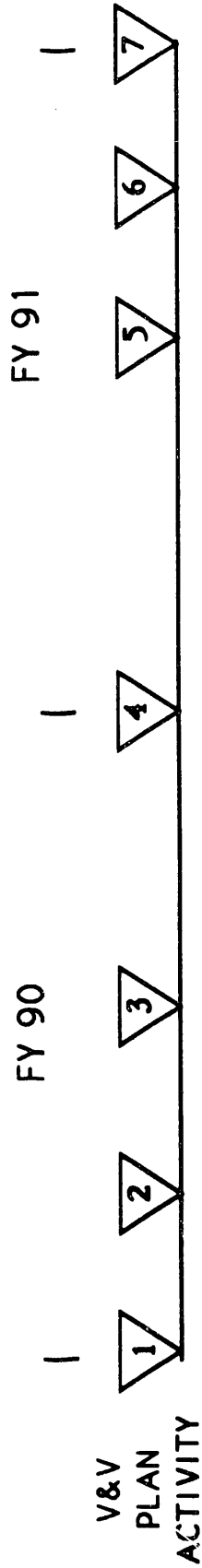
REFERENCES

1. Letter, "Technical Specification Parameter Justification", P.W. Dickson to R.E. Tiller, August 11, 1989.
2. Letter, "Approval of Technical Specification", P.W. Kasper to J. S. Moore, May 26, 1989.
3. A.A. Zagrodnik, QA Procedures for the Nuclear Reactor Technology and Scientific Computations Program Management Team, DPSTM-88-700-9, 1988

Figure 1
 Verification and Validation Plan Action Matrix
 Thermal Hydraulics Codes

	Basics	Theory	Experiments	Benchmarks	Conclusions	
FLOWTRAN-FI	User Manuals in Place Configuration Control Plan Code Portability	Appropriate Theory Theory Documented Coding Consistency Theory Verified Conceptually Theory Verified by Experiments Theory Documentation Adequate	Tests in Experimental Facilities Tests in Operating Facilities Data from Operating Facilities Test Data Documented Appropriate Data Quality Validation Performed Validation Documentation Adequate	Benchmark Requirements Identified Similar Code Comparison Exact Solution Comparison Industry Benchmark Comparison Comparisons Documented Benchmark Documentation Adequate	Verification review Verification Completed	Standard Set of Test Problems Validation/Benchmarking Review Validation/Benchmarking Completed
FLOWTRAN-TF						
FLOWZONE						
LLAP						
PIPEFLOW						
PLENUM						
RELAP5						
SCALEUP						
TRAC/MOD1						

Figure 3
**SCHEDULE FOR VALIDATION AND VERIFICATION PLAN
 ACTIVITIES**



Milestones

1. Complete a Verification and Validation Plan (Nov 15, 1989)
2. Complete Computer Code Configuration Control Plan (Jan 15, 1990)
3. Source Material and Activity Assessment for each code completed (Mar 30, 1990)
4. Basics and Theory Section of Verification Completed (Sep 30, 1990)
5. Verification Review Completed
 Experimental and Benchmark Sections of Validation Completed (Mar 30, 1991)
6. Standard Set of Test Problems in place and Documented
 Validation Review Complete (Jun 30, 1991)
7. Review Comments resolved
 Validation and Verification Activities Finished (Sep 30, 1991)

Code Descriptions

This APPENDIX contains code descriptions for the codes to be included in this verification and validation effort:

AA3
FLOWTRAN-FI
FLOWTRAN-TF
FLOWZONE
GILDA (see TRIMHX/GRIMHX/GILDA)
GLASS
GRIMHX (see TRIMHX/GRIMHX/GILDA)
HMTABLE
ICG
JASON
LLAP
MARCO
PIPEFLOW
PLENUM
PORAD
RELAP5
SCALEUP
SHIELD
TRAC/MOD1
TRIMHX/GRIMHX/GILDA
WIGGLE

AA3

CODE DESCRIPTION

AA3 is a coupled neutronics-thermal hydraulics calculation of various accident scenarios. The neutronics routines in AA3 are calculated using point kinetics equations. The hydraulics, developed specifically to SRS reactors, are modeled using techniques described in DPSTM-120. Flux shape changes are represented using independently determined "peaking factors".

APPENDIX

DATABASE DESCRIPTION

Cross sections are initially generated using GLASS. Histogram groupings of powers are supplied from GRIMHX results.

CODE APPLICATIONS

This code can be used to establish transient protection and confinement protection limits. The reactor core, coolant loop and protective instrument system are all modeled. Steam generation rates and steam pressure under the top shield and plenum are computed.

HISTORICAL DEVELOPMENT

AA3 was initially developed as the point kinetics code, BURP, written by Ed Bailey in the early 1960's. Thermal hydraulic effects were later added, and this code became known as AA3.

FLOWTRAN-FI

CODE DESCRIPTION

FLOWTRAN-FI is a single phase code that uses time-dependent pressure boundary conditions from TRAC output or another system code to predict onset of flow instability (OFI). This code uses implicit finite-difference methods to solve the partial differential equations that describe the conservation of mass, momentum and energy for a single phase fluid. FLOWTRAN-FI also contains correlations that account for the heated wall effect and the increased roughness and reduced flow area caused by nucleate boiling. This code cannot simulate fully developed boiling. FLOWTRAN-FI uses a detailed numerical model of a SRS reactor MK 22 assembly to simulate thermal-hydraulic conditions during normal and abnormal operating conditions. FLOWTRAN-FI also solves three-dimensional heat conduction equations that model heat transfer in the fuel tubes and target tubes. The power transient is an input to the code.

DATABASE DESCRIPTION

FLOWTRAN-FI is a self-contained code that does not require accessing external databases or data libraries. All material property and data correlations used by FLOWTRAN-FI are embedded within the code. The input deck contains all other information required to run the code such as power distribution functions and problem geometry.

CODE APPLICATIONS

SRL uses FLOWTRAN-FI to compute assembly effluent temperature limits for a variety of SRS reactor accident scenarios, including the DEGB LOCA, Pump Shaft Break, Gang Rod Withdrawal and Small (Bellows) Break LOCA. SRS reactor operators use these assembly effluent temperature limits to control reactor power during operation. (If ECS limits are lower, then the assembly effluent temperature limits computed by the FLOWTRAN-TF code will determine the reactor power limits.)

HISTORICAL DEVELOPMENT

SRL developed the FLOWTRAN-FI code in 1987 and 1988 to fill the need for a detailed thermal-hydraulic code to compute effluent temperature limits. The thermal-hydraulics codes used to set assembly effluent temperature limits prior to FLOWTRAN-TF were much cruder with respect to the governing equations, the assembly geometry and the numerical methods. The Nuclear Engineering Section (NES) of SRL developed the entire code and has written a detailed user's manual. NES has published the results of two benchmarking studies and is working on several more, which will be based on experimental data from tests which include rigs with prototypic geometry, flows, pressures and heat fluxes.

FLOWTRAN-TF

CODE DESCRIPTION

FLOWTRAN-TF is a two-phase, two-component code that uses time-dependent pressure boundary conditions from RELAP5 output or

APPENDIX

another system code to predict the occurrence of thermal excursions in Mark 22 fuel assemblies under mixed air-water flow conditions. Such conditions are expected in SRS reactors only during the later stages of a DEGB LOCA, when most of the moderator has drained from the reactor vessel and the Emergency Cooling System (ECS) has started to supply light water to the core. FLOWTRAN-TF uses implicit finite-difference methods to solve the seven partial differential equations that describe the conservation of mass, momentum and energy for liquid water and for a mixture of steam and air. FLOWTRAN-TF also contains correlations that model the interfacial transfer of heat, momentum and mass between slugs of water and air-steam mixtures. This code can simulate fully developed boiling, counter-current flow and flooding. FLOWTRAN-TF uses a detailed numerical model of a SRS reactor MK 22 assembly to simulate thermal-hydraulic conditions during the DEGB LOCA. FLOWTRAN-TF also solves three-dimensional heat conduction equations that model heat transfer in the fuel tubes and target tubes. The power transient is an input to the code.

DATABASE DESCRIPTION

FLOWTRAN-TF is a self-contained code that does not require accessing external databases or data libraries. All material property and data correlations used by FLOWTRAN-TF are embedded within the code. The user supplied input deck contains all other information required to run the code such as boundary conditions, power distribution parameters, and problem geometry.

CODE APPLICATIONS

SRL uses FLOWTRAN-TF to compute assembly effluent temperature limits for the DEGB LOCA. SRS reactor operators use these assembly effluent temperature limits to control reactor power during operation. (If Flow Instability limits are lower, then the assembly effluent temperature limits computed by the FLOWTRAN-FI code will determine the reactor power limits.)

HISTORICAL DEVELOPMENT

SRL is developing the FLOWTRAN-TF code to fill the need for a detailed thermal-hydraulic code to compute effluent temperature

limits when air is present in the assemblies. Prior to the mid-1980's SRS did not compute power limits for the DEGB LOCA, so air-water flows were not a concern in the limit-setting process. The thermal-hydraulics codes used to set assembly effluent temperature limits prior to FLOWTRAN-TF were much cruder with respect to the governing equations, the assembly geometry and the numerical methods. The Nuclear Engineering Section (NES) of SRL is developing the entire code and is writing a detailed user's manual. NES is performing experiments that will provide the data required for benchmarking of FLOWTRAN-TF. These experiments will generate data from test rigs with prototypic geometry, flows, pressures and heat fluxes. NES will finish coding and benchmarking the initial version of FLOWTRAN-TF by April, 1990.

FLOWZONE

CODE DESCRIPTION

The FLOWZONE code uses algebraic equations, simple theoretical models, and correlations to design the assembly flows of a reactor charge. The algorithm seeks to optimize an orificed reactor flow to the power distribution.

No feedback is provided from the remaining portion of the limits system and only 16 flowzones are allowed. FLOWZONE is executed in tandem with the PLENUM code, which generates process water plenum supply curves for a given core loading configuration.

DATABASE DESCRIPTION

Both thermodynamic and thermal-hydraulic correlations are employed to calculate core pressure drops and other fluid properties. FLOWZONE also receives input from the PLENUM computer code.

CODE APPLICATIONS

This code determines the zone average assembly flows for a given charge design. This allows for the matching of power distributions to flow, which is accomplished (mechanically) through the use of orificing. It outputs the orifices required to obtain the desired flow.

APPENDIX

It also gives flow test station predictions, which are necessary for the operation of a fuel cycle. Use of this code allows for increased coolant usage efficiency, and consequently a higher allowable reactor power.

GLASS

CODE DESCRIPTION

The (Generalized Lattice Analysis Sub-System) GLASS code is a thermal reactor physics code essential for evaluating lattice cell characteristics and input to data records for other codes. The GLASS computer code exists as a subset of the JOSHUA system, and solves the multi-group Boltzmann transport equations for a lattice cell or repeating cluster of cells, using integral transport theory. The lattice description is input by using "templates", which guide the user interactively through each input step. Boundary conditions may be either reflective or periodic. Certain regions within a problem may be decoupled from each other (explicitly) by conserving cosine-directed currents across their boundaries.

GLASS has several options for performing calculations. The most commonly used transport option is integral transport theory (using collision probabilities and response matrices), although a second option (Monte Carlo) is available for calculational cross comparisons. The resonance calculations in GLASS use the Nordheim Integral Treatment (NIT). The MUFT NIT treatment is employed for the epithermal energies, and the THERMOS approach is applied to the thermal energy groups.

DATABASE DESCRIPTION

The cross section data libraries available to GLASS include a spectrum independent 84-group library and a heavy water spectrum weighted collapsed 37-group library. The cross sections in the 84-group library were obtained from the Evaluated Nuclear Data File, Version B (ENDFB/III and IV). Data for U-238 was originally taken from ENDFB/III, but has since been modified for SRS use. Data for U-235, Li-6, and Al were extracted from ENDFB/IV. However, the value for the average number of neutrons per fission (ν) for U-235 has been updated to agree with ENDFB/V. Most lattice

calculations are performed using the 37-group library due to the reduced computer execution times.

CODE APPLICATIONS

GLASS is used to calculate the multi energy group fluxes and reaction rates in hexagonal single or supercell configurations for SRS fuel designs. Depletion calculations are possible that permit estimation of special isotope production. Reactivity effects can be computed to establish reactivity coefficients. GLASS is also employed to produce few-group cell averaged nuclear cross sections, used in two dimensional (2D) diffusion theory reactor analysis codes, such as GRIMHX (for steady-state) and TRIMHX (for transient analysis). The code provides most of the basic reactor physics information pertinent to certain Technical Specification parameters.

HISTORICAL DEVELOPMENT

The development of the current GLASS system started in 1965 with the development, by H.C. Honeck, of the THERMOS computer code, which analyzed neutrons in thermal energy ranges. The MUFT (NIT) treatment was then added to THERMOS for neutron slowing down computations. The coupled codes evolved into the HAMMER system, which was the first SRS calculational tool used to correlate with experimental lattice data. By 1971, HAMMER had evolved into the first version of GLASS, which used ENDFB/III cross sections for the fissile isotopes. Subsequent improvements have been made to both the cross section data, and the numerical methods involved in the resonance treatment. A Monte Carlo option was added, so that certain problems could be evaluated using both integral transport theory and stochastic methods. In the early 1970's work was initiated to develop the JOSHUA operating system to automate many of the I/O sequences between different computer codes, and GLASS was included in this system.

HMTABLE

CODE DESCRIPTION

The code models the SRP assemblies post shutdown fission product

APPENDIX

decay power histories according to the relations given in the American National Standard for Decay Heat Power in Light Water Reactors, ANSI/ANS-5.1-1979.

CODE APPLICATIONS

The code generates standard SRP reactor assembly fission product decay heat powers as a function of assembly exposure and decay time. This module does not interface with any other code modules.

The code is valid for calculating fission product decay heat powers for Mark 16, Mark 31, Mark 22 and Mark 15 assemblies.

ICG

CODE DESCRIPTION:

ICG is a data processing computer code which fits polynomial coefficients for cross sections and other data to up to seven independent variables.

DATABASE DESCRIPTION

Input data are usually provided (through other interfacing and reformatting computer codes, such as ICGLOAD) from the GLASS code system.

CODE APPLICATIONS

ICG is used to correlate lattice physics computer code results into fitted polynomial coefficients, which are then provided to the JASON system library, for later GRIMHX and TRIMHX analyses.

JASON

CODE DESCRIPTION

JASON calculates the time-dependent physics characteristics of a reactor charge. This code combines a two group, two-dimensional

diffusion calculation using GRIMHX with a burnup calculation. It processes the JASON library created by ICG in order to obtain cross sections. These cross sections are reformatted for use in the GRIMHX code. The GRIMHX output is then edited to give a power map and a list of isotope concentrations. Also edited are the buckling held in control rods and the radial statistical weights. The information provided by this code is essential to the design of a reactor charge.

JASON provides a diffusion theory solution of the neutron flux over the entire reactor. It begins with individual assembly cross sections named by the user and found in the JASON libraries. GRIMHX does the diffusion part of the calculation. The statistical weights are calculated using the flux squared method.

DATABASE DESCRIPTION

Same as for TRIMHX/GRIMHX

LLAP

CODE DESCRIPTION

The LOCA Limits Analysis Package (LLAP) has been created as a driver for the FLOWTRAN code which computes the operating power limit of a single SRS assembly subjected to a given thermal hydraulic transient. The purpose of LLAP is to streamline the limits calculational procedure by interfacing several input preparation codes and automating the data transfer. In addition, LLAP runs the limits analysis for the entire charge as a single job. In principal, no user intervention is required to generate the complete set of limits.

DATABASE DESCRIPTION

There is no external database or library which LLAP accesses.

CODE APPLICATIONS

LLAP is executed to perform FLOWTRAN calculations. Applications of FLOWTRAN are addressed in its own description.

APPENDIX

MARCO

CODE DESCRIPTION

MARCO (MARgin of COntrol) executes the diffusion theory codes GRIMHX and GILDA to calculate reactivity margins.

DATABASE DESCRIPTION

Input data libraries for MARCO may be traced to GLASS, since the neutronics modules GRIMHX and GILDA use cross sections which are generated by GLASS and processed through ICG. In addition, ICG produces fitted coefficients to represent the GLASS results.

CODE APPLICATIONS

MARCO is employed to evaluate the control system margin for SRP reactors, given a particular control rod pattern. The margin is defined as the difference in the calculated reactivity of the reactor with all the rods in and the reactivity of the reactor at critical state. MARCO also gives statistical weights (importances) of the different control rods. The statistical weights can be determined using the adjoint functions. The calculations are performed for the cold clean (no xenon or samarium) conditions. This amounts to the most conservative assumption because it gives the smallest margin of control.

PIPEFLOW

CODE DESCRIPTION

The PIPEFLOW code is a nationally known hydraulic code used to analyze steady state pressure and flow in pipe distributions. The basis of the code is a direct solution of the pipe system of hydraulic equations using a linearization scheme and sparse matrix methods to handle the co-linear terms in the energy equations.

DATABASE DESCRIPTION

Empirical data contained within the code include loss coefficients for different types of pipe fittings, as well as other correlations

adequate for the entire spectrum of Reynolds numbers, from laminar through turbulent flow regimes.

CODE APPLICATIONS

PIPEFLOW is used for a variety of problems at SRS, including the development of a cooling water and emergency cooling system (ECS) model for use in the SRS Reactor Training Simulator, as well as to calculate cooling water system leak flows for the Probabilistic Risk Assessment (Level 1 PRA) for the SRS reactors.

HISTORICAL DEVELOPMENT

The May 1983 version of PIPEFLOW, purchased by the Engineering Department, includes several modifications requested by SRL to improve hydraulic modeling of the Reactor Training Simulator. The most significant modifications include:

- 1) calculation of "tee" and "wye" fitting hydraulic losses (which are a function of branch flow splits and area ratios) and
- 2) calculation of other fitting (e.g. valves, elbows, etc.) hydraulic losses by the more accurate "two-K" method, instead of the standard "number of velocity heads" or "equivalent pipe lengths" methods. The "two-K" method is valid for the whole spectrum of Reynolds numbers, whereas the other standard methods are valid only for Reynolds numbers corresponding to fully developed turbulent flow. The August 1987 version of PIPEFLOW includes modifications by SRL in the summary printout to account for key data related to the fourth ECS addition system.

PLENUM

CODE DESCRIPTION

This code computes the process water plenum supply curve for a given reactor charge configuration.

APPENDIX

DATABASE DESCRIPTION

The plenum head available curve is generated by summing up the pressure drops from the balance of the hydraulic system and then subtracting this from the pump supply pressure.

CODE APPLICATIONS

PLENUM output is used by the FLOWZONE and SCALEUP computer codes. The plenum supply curve is used by the reactor control computers to calculate pump flow, which is used to evaluate reactor goal power, as well as on-line evaluation of effluent limits.

PORAD

CODE DESCRIPTION

The PORAD code processes output data from JASON and reformats data for input into other codes.

DATABASE DESCRIPTION

This interface code links JASON output with FLOWZONE input.

CODE APPLICATIONS

PORAD reads the JASON power map and converts it to a format which can be read by POWERMAP, FLOWZONE and FLOOD84. It also sets all central powers equal to 1 and normalizes the other powers. It analyzes reactor power data for flatness, peaking and assembly power variations during a subcycle. The purpose of this flattening is to reduce the number of flowzones.

RELAP5/MOD2.5

CODE DESCRIPTION

RELAP5/MOD2.5 is a pressurized water reactor (PWR) system transient analysis code that can be used for simulation of a wide variety of PWR system transients of interest in light water reactor

(LWR) safety. It is an extension of RELAP5/MOD2 which includes some error correction and certain additional models and options not included in the earlier version. The primary system, secondary system, feedwater train, system controls, and core neutronics can be simulated. The code models have been designed to permit simulation of postulated accidents ranging from large break loss-of-coolant accidents to accidents involving the plant controls and fuel system. Transient conditions can be modeled up to the point of fuel damage.

The RELAP5/MOD2.5 code solves six basic nonhomogeneous, nonequilibrium field equations for six dependent variables -- pressure (P), specific internal energies (U_g and U_f), void fraction (α_g), and velocities (v_g and v_f). The equation set is extended to include the presence of a noncondensable component and a liquid solute component through the addition of a mass conservation equation for each of the additional components.

The field equations for the two-fluid model consist of two phasic continuity equations, two phasic momentum equations and two phasic energy equations. The equations are recorded in differential streamtube form in terms of time and volume average dependent variables, with time and one space dimension as independent variables.

Structural components such as reactor vessels and piping can be modeled as heat structures which communicate with the the primary or secondary coolant. The one-dimensional heat conduction equation is solved for each heat structure, with a variety of boundary conditions available to the user. Heat sources can be input by the user or calculated with a point kinetics model that includes reactivity feedback from fuel and water temperature, control rods, and boron (or other soluble poisons).

User options in RELAP5/MOD2.5 are extensive. A variety of special component models allow the user to construct a detailed representation of the primary, secondary, and control systems in a reactor plant. Input is free format and keyed by card number. Mass, momentum, and energy boundary conditions are flexible and can be set by the user through standard input.

APPENDIX

DATABASE DESCRIPTION

Material properties are taken from the MATPRO library of properties. This library was generated by the Idaho National Engineering Laboratory to describe reactor materials under a wide range of physical conditions, including the severe conditions expected during reactor operating and accident situations.

Steam and water properties are based on ASME Steam Tables for light water properties, and upon data from the University of British Columbia for heavy water. The tables of heavy water properties were generated by using a NASA code as modified by the Helmholtz free energy equation and the saturation equation taken from the Canadian data of Hill et al.

CODE APPLICATIONS

RELAP5/MOD2.5 code uses include analytical support for NRC rule making, licensing audit calculations, evaluation of accident mitigation strategies, and experiment planning and analysis. Specific applications of this capability have included analytical support for the loss-of-fluid test (LOFT), Power Burst Facility (PBF), ACRR, MIST, ROSA-IV, and NRU experimental programs, as well as simulations of transients that lead to severe accidents, such as loss of coolant, anticipated transients without scram (ATWS), and operational transients in LWR systems.

Input to the code is through card images containing geometric information describing discrete volumes and connecting junctions used to represent the reactor or experimental facility being modeled. Output is in the form of printed output (either paper or microfiche), graphical output, and tabular output recorded on computer files for input to post-processing routines. The output forms are controlled by the user through standard input. Input to codes using the RELAP5 results is most easily accomplished through post-processing routines that access tabular data generated by RELAP.

HISTORICAL DEVELOPMENT

The series of RELAP codes began at the Idaho National Engineering Laboratory with RELAPSE (REactor Leak And Power Safety Excursion), which was released in 1966. Subsequent versions of this

code were RELAP2, RELAP3, and RELAP4, in which the original name was shortened to Reactor Excursion and Leak Analysis Program (RELAP). All of these codes were based on a homogeneous equilibrium model (HEM) of the two-phase flow process. The latest version of this series was RELAP4/MOD7, which was released to NESC in 1980.

In 1976, the development of a nonhomogeneous nonequilibrium model was undertaken for RELAP4. When it became apparent that a total rewrite of the code would be required to accomplish this goal, the RELAP5 project began. This was the fifth in the series of computer codes that was designed to simulate the transient behavior of LWR systems under a wide variety of postulated accident conditions. The principal new feature of the RELAP5 series was the use of a two-fluid nonequilibrium nonhomogeneous hydrodynamic model for transient simulation of the two-phase behavior. RELAP5/MOD2 employs a full nonequilibrium, six equation, two-fluid model. Use of the two-fluid model eliminates the need for the RELAP4 submodels such as the bubble rise and enthalpy transport models, which were necessary to overcome the limitations of the single fluid model.

RELAP5 was originally developed to run on a Control Data 7600 series computer. In recent years, it has been modified to run on a CRAY XMP. It is written in FORTRAN and has had most of the machine specific coding removed to make it portable. The code can be configured to run under a variety of operating systems through options set during compilation. The current version being used at SRL is RELAP5/MOD2.5, Version 3b3, with modifications that have been agreed upon with the INEL, documented, and saved.

SCALEUP

CODE DESCRIPTION

This code computes the expected zone averaged flows and monitor pin pressure drops as well as hot flows and pressure drops which are to be measured in the flow test station. The code reads supply curves from PLENUM and iterates until the zone flow is correct.

APPENDIX

The SCALEUP code uses a successive substitution technique to derive the zone average flows, and is limited to 16 flowzones.

DATABASE DESCRIPTION

SCALEUP receives its input data sets from the PLENUM code.

CODE APPLICATIONS

Thermal-hydraulics operating limits are derived partly from the calculated core pressure drops, zone average flows, and flow reduction tolerances output from the SCALEUP computer code.

SHIELD

CODE DESCRIPTION

The SHIELD system is composed of a set of calculational routines which have been created to execute several types of shielding calculations (isotopic composition, spontaneous radiation source, neutron, and cluster geometry). The following is a description of the individual components included in the shield system:

SHIELD: This is the driver module for the SHIELD system. It carries out the calculational sequence specified by the user. It also provides a common base for the calculations.

CONVERGE: This module is used to test for convergence of the isotopic number densities in modeling coupled flow sheets with PROCESS module. It can be used to check on the convergence when a PROCESS calculation is being executed. It can also attempt to speed convergence. The convergence criterion is based on a percentage of change in the isotopic concentration. The convergence is enhanced through the use of a geometrical extrapolation technique.

FIPROD: This module is a point depletion module. It passes the flux history, initial isotopic cross sections, and initial concentrations to the module FPCALC.

FPCALC: This module uses either of two methods to determine isotopic concentration as a function of time. It can use the matrix

exponential method or the linear chain method. Tests have indicated that both these methods yield similar results.

PROCESS: This module calculates the isotopic inventory changes due to chemical or mechanical changes in the fuel cycle. This module can solve simple or detailed processes to the degree of accuracy required by the user. An iterative method is used to solve the mass balance equations. This technique requires less storage than using a matrix method and provides the same accuracy.

SNONE, SNTWO: These are the one- and two-dimensional neutronic codes which can be executed with SHIELD. They are used primarily to determine neutron and photon flux distributions in a given one- and two-dimensional geometry. They can also perform buckling, eigenvalue, and fixed source problems and boundary searches.

Both of these codes use the discrete ordinates, S_n , transport method to solve the transport equation. These codes are adaptations of DTF-IV (one-dimensional) and TWOTRAN2 (two-dimensional). The "diamond difference" method is employed.

XPREP2: This module prepares mixture cross sections, i.e., it combines several elemental or isotopic cross sections to form material cross sections. These material cross sections can have either the same energy spectrum or a collapsed structure. The collapsing spectrum may be specified by the user or can be obtained from lower dimensional S_n calculations. The collapsing of the cross sections is handled using flux weighting.

RADSOR: This module calculates the spontaneous neutron and photon source (particles/sec) resulting from all decay processes. The code obtains this by summing the decay spectrums of all the different isotopes multiplied by their concentrations (generated in FPCALC or specified by the user) and half lives. A vector containing the isotopic concentrations is scaled by the decay spectra and decay rate.

SHLDEDIT: This module controls the editing of data from the shield calculations and from data stored in the shield data set records. General, process, cross section, geometry, radiation field and dose edits can be performed.

APPENDIX

DOSE: This module calculates the biological dose rate field and material heating from flux-to-dose rate factors, KERMA factors and a flux generated by a neutron-photon transport (e.g. SNONE or SNTWO).

CODE APPLICATIONS

SHIELD is comprised of a set of modules which can perform the following applications:

1. One- and two-dimensional neutron transport
2. Cross section mixing and collapsing
3. Process modeling
4. Isotopic inventory calculations
5. Personnel dose calculations
6. A plethora of edits which can be plotted or output in a tabular form.

The system of routines has the capability of modeling fuel storage facilities, shipping casks, fuel process facilities, and waste disposal areas. This spans the whole range of the out-of-reactor period of the radioisotopes.

TRAC-PF1/MOD1

CODE DESCRIPTION

The TRAC-PF1/MOD1 system code (hereafter referred to as TRAC) was developed by Los Alamos National Laboratory (LANL) over a period of many years to solve the coupled set of partial differential equations that describe the thermal-hydraulic behavior of fluid coolants in pressurized nuclear reactors. This system of equations includes conservation equations for the liquid energy, mass and momentum. The developers of TRAC assumed that the condensible and noncondensable gases are well-mixed, and thus can be modeled by energy and momentum equations for the mixture.

TRAC solves separate mass conservation equations for noncondensable and condensable gases. In addition, TRAC uses one and two dimensional heat conduction equations to describe the flow

of energy in the fuel and the structural components of the reactor. TRAC models generation of the nuclear power in the reactor core either by direct input of the power transient or by solution of point-reactor kinetics with reactivity feedback.

Like all thermal-hydraulics codes, TRAC uses empirical relationships between state variables to model heat, mass and momentum transfer between solid boundaries and the coolant and between the different fluid phases and components. Since LANL developed TRAC for pressurized reactors, some of these relationships may not be valid for the SRS reactors. For this reason, SRS is benchmarking TRAC against hydraulic data taken in SRS reactors.

TRAC solves the equations described above with finite-difference, semi-implicit numerical methods. TRAC uses a detailed numerical model of the SRS reactors which contains about 1400 nodes. The numerical simulation of transients, such as DEGB LOCA's, starts with steady-state solutions that closely approximate normal operating conditions.

DATABASE DESCRIPTION

The TRAC code, as implemented at SRL, does not use any external databases or libraries.

CODE APPLICATION

SRS uses TRAC to simulate reactor response to several accident scenarios, i.e., DEGB LOCA, Pump Shaft Break Accident, Gang Rod Withdrawal Accident and Small (Bellows) Break LOCA. These simulations provide boundary conditions for the assembly codes (FLOWTRAN-FI and FLOWTRAN-TF) that compute assembly effluent temperature limits for SRS reactors. (The assembly effluent temperature limits determine the reactor power level.) Principal inputs to TRAC are initial power, break location and power transient data. The important outputs are the plenum and tank bottom pressure transients, because they serve as the boundary conditions for the FLOWTRAN-FI code calculations of assembly effluent temperature limits.

APPENDIX

HISTORICAL DEVELOPMENT

TRAC was developed by LANL during the 1970's and 1980's for power reactor accident analysis. SRL brought TRAC to SRS in the mid-1980's to upgrade its LOCA analyses. TRAC is an industry standard code, and extensive documentation is available. LANL developed TRAC for use on supercomputers, usually CRAY's.

SRL has benchmarked TRAC against existing SRS reactor hydraulic data and is now benchmarking TRAC against the 1989 L Reactor Hydraulics Test data.

SRL uses the released version of TRAC-PF1/MOD1, Version 14.3, with Fixpack updates. SRL has modified the correlations for single phase wall friction to better represent the transition from laminar to turbulent flow conditions. The code was also modified to permit it to run under the UNICOS operating system on the SRS CRAY X-MP supercomputer.

TRIMHX/GRIMHX/GILDA

CODE DESCRIPTION

TRIMHX and GRIMHX are 3D (hex-z geometry) diffusion theory computer codes which are used to calculate both transient and static (respectively) neutron flux distributions. Both of these codes are based upon finite difference approximations to the neutron diffusion equations in a few-group formulation. Numerical methods employed in GRIMHX include a successive line-relaxation, block inversion technique for the inner iteration, and a coarse-mesh rebalancing technique with fission source over-relaxation for the outer iteration. GRIMHX and TRIMHX solve the neutron diffusion equations over the entire reactor, while GILDA solves these equations for a single lattice cell.

Time integration options in TRIMHX include performing time steps using either a direct time integration, or by an exponential transform method, which uses a separation of variables technique to isolate the spatial-temporally dependent flux response into spatially dependent and time dependent responses. This approach increases the accuracy of the calculation.

Both GRIMHX and TRIMHX are variations of the same computer code methodology. Both perform inner iterations to solve for flux distributions in the same manner. However, whereas GRIMHX performs inner iterations in order to perform a single "source" iteration (in order to calculate k-effective), TRIMHX performs inner iterations in order to march a spatial flux shape from one time step to the next. TRIMHX also needs an initial flux shape (steady-state) from which to initiate a transient.

The GRIMHX and TRIMHX codes exist as parts of the JOSHUA system.

DATABASE DESCRIPTION

The cross sections and coefficients which are used by the GRIMHX and TRIMHX codes are produced by the GLASS lattice physics computer code. Transient calculations are modeled as being a function of many state variables, thus requiring a large number of GLASS calculations. These GLASS calculations result in a large number of cross section sets, which are then correlated, and these fitted coefficients are then input into TRIMHX.

CODE APPLICATIONS

GRIMHX is used for 3D (2D is optional) flux calculations for an entire reactor. TRIMHX is used for the evaluation of transient responses to certain homogeneous and localized reactivity perturbations.

HISTORICAL DEVELOPMENT

In the early 1970's work was initiated to develop the JOSHUA operating system, which was intended to automate many of the I/O sequences between different computer codes, and GRIMHX/TRIMHX was included in this system.

WIGGLE

CODE DESCRIPTION

WIGGLE is a one-dimensional, two-group transient diffusion theory calculation that was written to provide axial power profiles during a safety rod scram following a LOCA. It involves a simultaneous

APPENDIX

solution of $N+2$ equations, where N is the number of precursors and the other two are the two groups of the neutron flux. The time dependence of the flux is formulated implicitly; the time dependence of the precursor densities is solved for explicitly. The code is explained in more detail in DPST-87-511; a listing of the source code is also provided in the reference.

DATABASE DESCRIPTION

All input data is provided to the code by LLAP, which acts as a driver for the code. The axial spatial subdivision is based on the detail of the power profile needed and the cross sections are taken from GLASS calculations (uncorrelated) of appropriate lattices. The sources of all of the data are documented in DPST-87-511.

CODE APPLICATIONS

The code was designed for a single application and is so used by the LLAP code. No stand-alone version exists.

END

**DATE
FILMED**

5 / 17 / 93

