30
11/31/86

M.L.P.

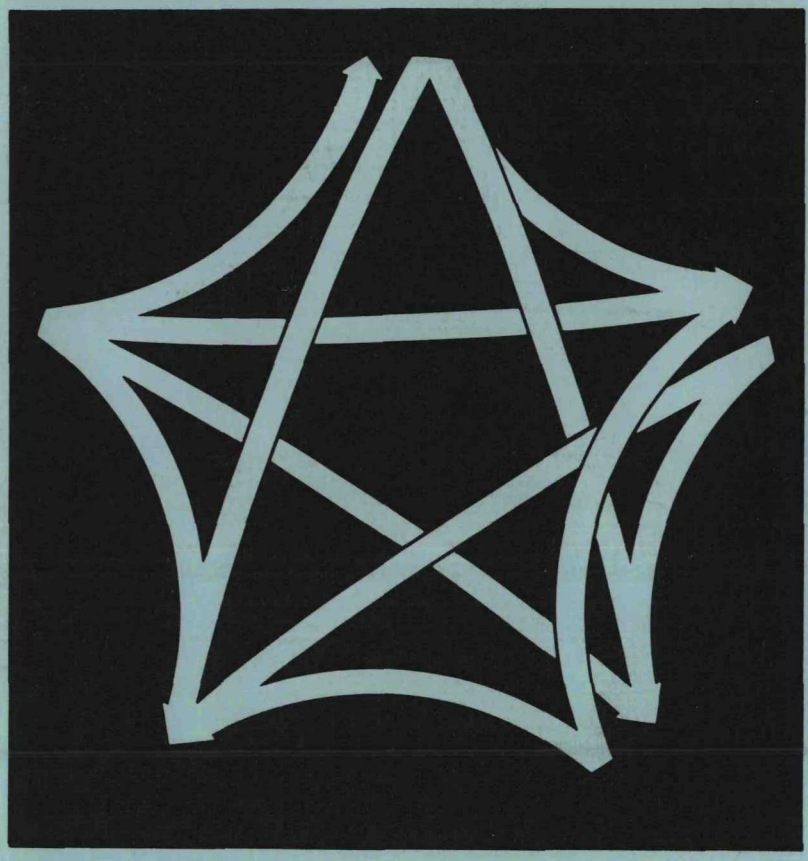
3 DIST

I-28799

DR 0040-X

PNL-5728
UC-71

WASTES:
Waste System Transportation and Economic Simulation - Version II Programmer's Reference Manual



November 1986

Prepared for the U.S. Department of Energy
under Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory
Operated for the U.S. Department of Energy
by Battelle Memorial Institute

PNL-5728

DISCLAIMER

PACIFIC NORTHWEST LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC06-76RLO 1830

Printed in the United States of America
Available from
National Technical Information Service
United States Department of Commerce
5285 Port Royal Road
Springfield, Virginia 22161

NTIS Price Codes
Microfiche A01

Printed Copy

Pages	Price Codes
001-025	A02
026-050	A03
051-075	A04
076-100	A05
101-125	A06
126-150	A07
151-175	A08
176-200	A09
201-225	A010
226-250	A011
251-275	A012
276-300	A013

DISCLAIMER

PNL-5728
UC-71

DISCLAIMER

PNL--5728

DE87 002838

WASTES: WASTE SYSTEM TRANSPORTATION AND ECONOMIC SIMULATION - VERSION II

PROGRAMMER'S REFERENCE MANUAL

M. E. Buxbaum[a]
M. R. Shay

November 1986

Prepared for
the U.S. Department of Energy
under Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory
Richland, Washington 99352

(a) Affiliated with Boeing-Computer Services, Inc., Richland, Washington.

SUMMARY

The WASTES Version II (WASTES II) Programmer's Reference Manual was written to document code development activities performed under the Monitored Retrievable Storage (MRS) Program at Pacific Northwest Laboratory (PNL), which is operated for the U.S. Department of Energy by Battelle Memorial Institute. The manual will also serve as a valuable tool for programmers involved in maintenance of and updates to the WASTES II code.

The intended audience for this manual are experienced FORTRAN programmers who have only a limited knowledge of nuclear reactor operation, the nuclear fuel cycle, or nuclear waste management practices. It is assumed that the readers of this manual have previously reviewed the WASTES II Users Guide published as PNL Report 5714 (Shay and Buxbaum 1986).

The WASTES II code is written in FORTRAN 77 as an extension to the SLAM commercial simulation package. The model is predominately a FORTRAN based model that makes extensive use of the SLAM file maintenance and time management routines.

This manual documents the general manner in which the code is constructed and the interactions between SLAM and the WASTES subroutines. The functionality of each of the major WASTES subroutines is illustrated with "block flow" diagrams. The basic function of each of these subroutines, the algorithms used in them, and a discussion of items of particular note in the subroutine are reviewed in this manual. The items of note may include an assumption, a coding practice that particularly applies to a subroutine, or sections of the code that are particularly intricate or whose mastery may be difficult.

The appendices to the manual provide extensive detail on the use of arrays, subroutines, included common blocks, parameters, variables, and files.

ACKNOWLEDGMENTS

The development of the WASTES model was jointly sponsored by the MRS Program Office at PNL and the Transportation Technology Center at Sandia National Laboratory. WASTES II contains significant enhancements to earlier versions of the model. These enhancements were primarily funded by the MRS Program for use in their integrated system cost and logistics studies. Many individuals contributed to the preparation and publication of this report. The following listing acknowledges the contributions of those involved in this effort.

Lead Engineer

M. R. Shay

Model Development Team[a]

M. E. Buxbaum

R. A. Heller

S. J. Ouderkirk

W. J. Suitt

(a) Affiliated with Boeing Computer Services Richland, Inc., Richland, Washington.

CONTENTS

FIGURES

TABLES

1.0 INTRODUCTION

The WASTES model was developed for use in analyzing the effects of various policy decisions, logistics considerations, and facility operating schedules on contemplated nuclear waste management system configurations. The model provides extensive capabilities for simulating the response of the waste management system under a variety of configurations and operational philosophies. WASTES uses discrete event simulation techniques to model the generation of commercial spent nuclear fuel, the buildup of spent-fuel inventories within various components of the system, and the transportation requirements for the movement of spent fuel throughout the system. The model is written in FORTRAN 77 as an extension to the SLAM commercial simulation language package (Pritsker and Pegden 1979).

In addition to pool storage and dry storage located at nuclear reactors, the WASTES model accepts up to a total of 10 waste processing facilities of four different types. The allowable types of processing facilities are federal interim storage (FIS), monitored retrievable storage (MRS), reprocessing plants, and repositories.

Reactor-dependent data supplied for use in the WASTES model is based on the PNL spent-fuel data base. This information includes reactor location, reactor type and transportation access, and historical and projected discharge data on the number of fuel assemblies, their fuel weight expressed in metric tonnage (metric tonnes uranium or MTU), and the expected exposure for each discharged spent-fuel assembly. Users have the option of providing their own data files.

The simulation may be driven by a combination of reactor- and/or destination-requested transfers. Reactor-driven transfers would occur when a reactor spent-fuel storage pool violates its full core reserve (FCR) storage margin or when a reactor is decommissioned. In either of these cases, the material requiring transfer is then shipped to facilities with available capacity. Destination-driven transfers occur when the annual capacity of a processing facility (e.g., an MRS facility or repository) will not be met by

reactor-driven transfers and spent fuel must be scheduled from reactors with non-critical storage needs or from upstream facilities in the waste management system.

Shipments within the system can be user-specified to occur optimally or proximally. Optimized shipping can be used when exactly two destination facilities of the same facility type are open for receipt of fuel. This algorithm will select allowable destinations for each fuel transfer so that the total shipping distance or total shipping costs in a given year will be minimized. If sufficient fuel does not exist to fully utilize both facilities, the model will attempt to fill the first facility's annual receipt rate. Proximity shipping fills the closest storage facility to the source of the spent fuel according to the shipment priorities. That approach results in sub-optimal routing of waste material but can be used to approximate an optimal shipping strategy when more than two processing facilities of the same type are available to receive waste.

The user supplies information on characteristics for shipping casks that can be used for truck or rail shipments. As part of the cask characteristics, the user specifies, for each facility, which casks can be used by the facility and whether the casks are limited to only shipping or receiving functions. Up to a total of 10 different casks may be characterized. This information is then used for determining allowable source/destination pairs and in estimating the transportation system costs.

Reports generated by WASTES include: annual inventories of spent fuel in each facility, annual shipment summaries between facilities, characterization of fuel received at each facility, and shipping summaries giving detail on costs, number of required casks, cask miles, etc.

2.0 GENERAL STRUCTURE OF THE CODE

WASTES is a FORTRAN 77 based discrete-event simulation model that utilizes SLAM simulation and file maintenance routines. SLAM maintains an event calendar that contains the time an event is to occur and information specific to the event to be modeled. The entries in the event calendar are sorted according to time with the lowest value being first. As simulated time progresses, SLAM calls the user-written event processing routine (subroutine EVENT), which performs the actions appropriate for the specific event taking place (usually via other user-written routines which are in turn called from EVENT). The simulation then advances to the next discrete instant in time and processes the new event.

The events that are recognized by WASTES are shown in Table 2.1.

TABLE 2.1. Events Processed by WASTES

Event Code	Type of Event	Subroutine Called
1	Reactor Discharge	RXDIS
2	Facility Fuel Solicitation	FAC
3	Change in Facility Capacities	CHGCAP
4	Change in Facility Priorities	CHGPRI
5	Reactor Decommissioning	DECOM
6	Reserved for Future Expansion	
7	Reserved for Future Expansion	
8	Change in Alternate Priorities	CHGALT
9	Change in Fractional FCR	none
10	Transshipment Enabled/Disabled	none

The three events that cause the processing of material movements are RXDIS, DECOM, and FAC. RXDIS occurs throughout the calendar year (year.00 to year.93) and results in the processing of reactor discharges as scheduled in the user-defined historical and projected reactor-discharge data files. DECOM attempts to unload spent fuel stored at decommissioned reactors after all reactor discharges in a given year have been processed. This unloading is accomplished by scheduling the decommissioning event to occur toward the end of each year

(year.95). Finally, the destination facilities attempt to fill unused annual capacity at the end of each year (year.99).

In addition to running the simulation itself, which SLAM initiates and continues via multiple calls to subroutine EVENT, SLAM also calls two other user-written subroutines. These subroutines perform initial set up of the simulation and generate the reports at the end of the simulation. These subroutines are respectively, INTLC and OTPUT.

2.1 SLAM QUEUES AND THE BATCH ARRAY

Each discharge that is read in from the discharge data file is referred to as a "batch" or "sub-batch." Accountability is maintained for each individual batch. At numerous times in the simulation it is necessary or advantageous to split a batch, thus creating two new sub-batches. Full accountability is also maintained for each of these new sub-batches. The minimum batch/sub-batch size is one fuel assembly; therefore, in the course of a simulation it is possible to maintain accountability for each individual fuel assembly. The items of information that are maintained in the batch array for each batch or sub-batch are shown in Table 2.2.

To minimize the computer storage requirements, the batch array stores this information in a compressed form. Whenever batch handling operations occur, the compressed information contained in the batch array for the fuel batch being processed is transferred into the "working batch arrays." These arrays are a pair of equivalenced arrays named WRKBAT and IWRKBAT. WRKBAT and IWRKBAT contain, respectively, the real and integer fuel batch information values after expansion. The array and index holding each batch variable is also shown in Table 2.2.

Those elements of the batch description that may change during the simulation are numbered 2,3,7,8. Note that the number of assemblies in the batch (Item 3) will change if the batch is split. The heat-generation rate of the batch (Item 7) is updated each time the batch is moved to a new location.

A unique queue is used to model each reactor pool and each facility in the simulation. A single queue is used to model the dry storage located at

TABLE 2.2. WRKBAT/IWRKBAT Array Contents

Working Batch Array Element	Fuel Batch Information Packet
WRKBAT(1)	Time of reactor batch discharge (yr, mo)
IWRKBAT(2)	ID of facility where currently stored
IWRKBAT(3)	Number of assemblies
WRKBAT(4)	Average weight of fuel per assembly (MTU/assay)
IWRKBAT(5)	Discharge exposure (MWd/MTU)
IWRKBAT(6)	Unique number to identify individual batches
WRKBAT(7)	Current heat rate (kW/MTU)
WRKBAT(8)	Time of arrival at current storage facility
WRKBAT(9)	ID of the reactor from which discharged
IWRKBAT(10)	Waste Type (PWR, BWR).

all reactors (NDRY). For efficiency reasons, a single queue also contains duplicates to all batches in all reactor pools. In this way, WASTES need only look in a single queue to determine the oldest batch in any of the reactor queues. At that time, each waste management facility and the generic pool and dry storage models (NPOOL and NDRY) are each assigned two queues. The first queue is used to perform age sorting while the second queue is currently reserved for future expansion of the model (heat sorting).

The reactor IDs and assigned queues are offset so that the lowest numbered queues are used by the waste management processing facilities and the generic NPOOL and NDRY queues discussed above. The value of this offset is represented by the variable ISTART and is calculated by the following formula:

$$\text{ISTART} = (\text{number of facilities} + 2) * 2.$$

If the simulation does not include the first reactor, an additional offset of IFIRSTR is calculated. In this way the first reactor to be modeled (regardless of what its reactor ID is) will be assigned to the queue immediately following NDRY.

Each entity in a queue contains two attributes: 1) the time of discharge for each source and 2) a pointer to the batch array. As each batch is moved, the entity is inserted in the appropriate place in the queue according to the

age of the fuel (LVF). By removing fuel from the top of each queue, the oldest material at each facility is selected. Use of a pointer allows a maximum of only two defined attributes (fuel age and a pointer to the batch array) to be required for each batch.

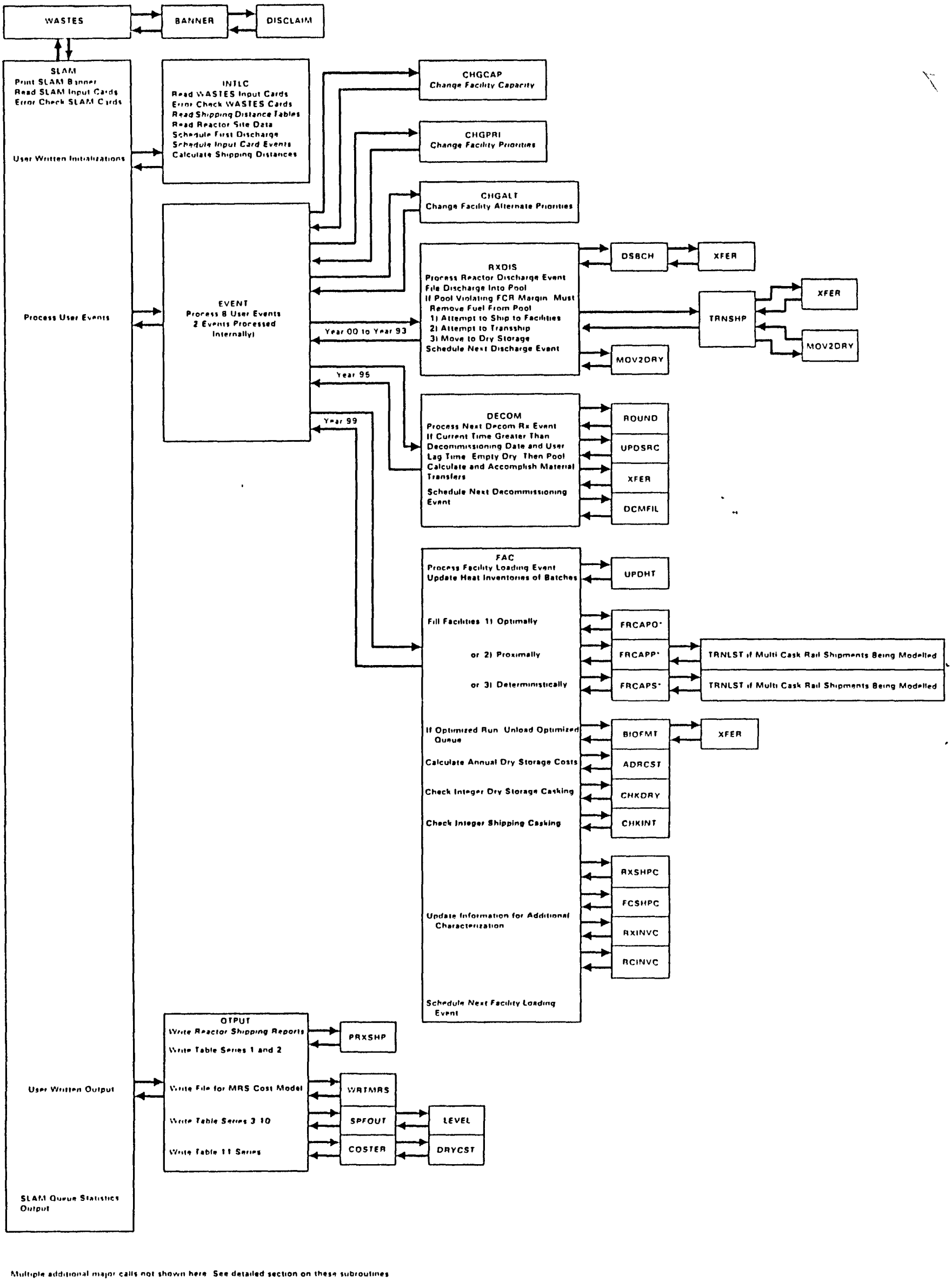
Several additional SLAM queues are used as working queues. These additional queues are discussed in Sections 4.2, 5.0, 6.4, and 6.6. Appendix D contains a complete listing of all queues used by WASTES.

2.2 GENERAL SEQUENCE OF EVENTS AS WASTES EXECUTES

Figure 2.1 shows a block flow diagram that illustrates the general sequence of events that occur as WASTES executes. The figure shows the sequence of subroutine calls but does not detail the actions within the various subroutines.

As execution begins, the WASTES banner is written to unit 6. SLAM then reads in the SLAM input cards and checks for errors. INTLC reads in the user-generated WASTES input cards, checks for errors, reads the reactor-site data file, calculates shipping distances, schedules the first reactor discharge, and performs multiple initialization and set-up activities for the WASTES run.

Typically at the beginning of the year (year.00) the events that change shipping priorities, facility capacities, or transshipment occur. The reactor-discharge events for the year occur at the time scheduled in the discharge data file (from year.00 to year.93). If, as a result of the discharge, the FCR margin of the pool is violated, the reactor ships or moves a sufficient amount of fuel from its pool to maintain the FCR margin. The reactor will first attempt to ship to any open facility that has available capacity to accept fuel and for which the reactor has acceptable fuel. If no facilities are open or none can accept the fuel, the reactor will then attempt to transship fuel if transshipment is allowed. As a final resort, the reactor will move the excess fuel into an at-reactor dry storage yard. Shipments to waste management facilities or transshipments to other reactor pools occur in integer (full) shipping cask quantities. Fuel movements from the pool to dry storage occur in integer (full) dry storage cask quantities.



Multiple additional major calls not shown here See detailed section on these subroutines

FIGURE 2.1. General Sequence of Events as WASTES Executes

At year.95, after all discharges for the year have occurred, WASTES attempts to move fuel from reactors that have been decommissioned. The subroutine that accomplishes this is DECOM.

Following that, at year.99, destination facilities will attempt to solicit enough acceptable fuel to fill any unused annual receipt capacity that the facility may have. This is accomplished through the subroutine FAC and, depending on the shipping algorithm being used, subroutine FRCAPO, FRCAPP, or FRCAPS. Since FAC executes every year and is the last major subroutine, a number of "housekeeping" operations are undertaken at this point (e.g., the annual shipping arrays are reset to zero, the annual loading capacities are reset for each facility, reports are written, and internal error-checking routines are called).

The sequence then repeats itself until the last year of the simulation is reached. When there are no more events to occur, SLAM will call OTPUT. OTPUT and the routines called (SPFOUT, LEVEL, COSTER) write the reports that give the results of the simulation.

3.0 SET-UP ROUTINES--SUBROUTINE INTLC

The primary subroutine that performs initialization activities before beginning the simulation is INTLC (see Figure 3.1). INTLC is called by WASTES before event processing begins. INTLC, in turn, calls several routines to perform specific portions of the setup activities. These routines, which are called RDPLNT, SCHDIS, and PROEVNT, will be discussed in more detail later. The following paragraphs detail those sections of the code whose actions or capabilities may not be immediately obvious from Figure 3.1.

The actions performed by INTLC are to read the WASTES input cards, to echo the options in effect for the run to the print file (unit 6), to set up WASTES variables and arrays according to the scenario being modeled, and to read the reactor-site data file.

Before attempting to read the shipping-distance data files, WASTES will determine if it can access the files. If they are not present or do not contain the facilities being modeled, WASTES will use an algorithm that is based on a great circle technique to estimate shipping distances. The first line of each of the data files read by WASTES contains a logical name and the file name. If these names do not match the file descriptions on the input card (reactor files) or the file names in BLOKDATA (shipping distance files), an error message will be printed and WASTES will not execute. In situations where multiple versions of a data file exist, this feature checks that the intended version is being used.

The categories for use in the age characterizations in the decommissioned reactor report are calculated in INTLC. The decommissioned reactor report will use up to three ages to characterize the ages of fuel at the reactors and the number of years the reactor is past its decommissioning date. These ages are the lowest three monotonically increasing ages read from the DECOM, AGETO, and DECOMAGE cards. Thus, it is possible to specify age acceptance for facilities that are not being modeled in the simulation to obtain more resolution on the decommissioned reactor report.

The prior array is initialized so that, by default, FIS facilities have the highest priority to unload their fuel.

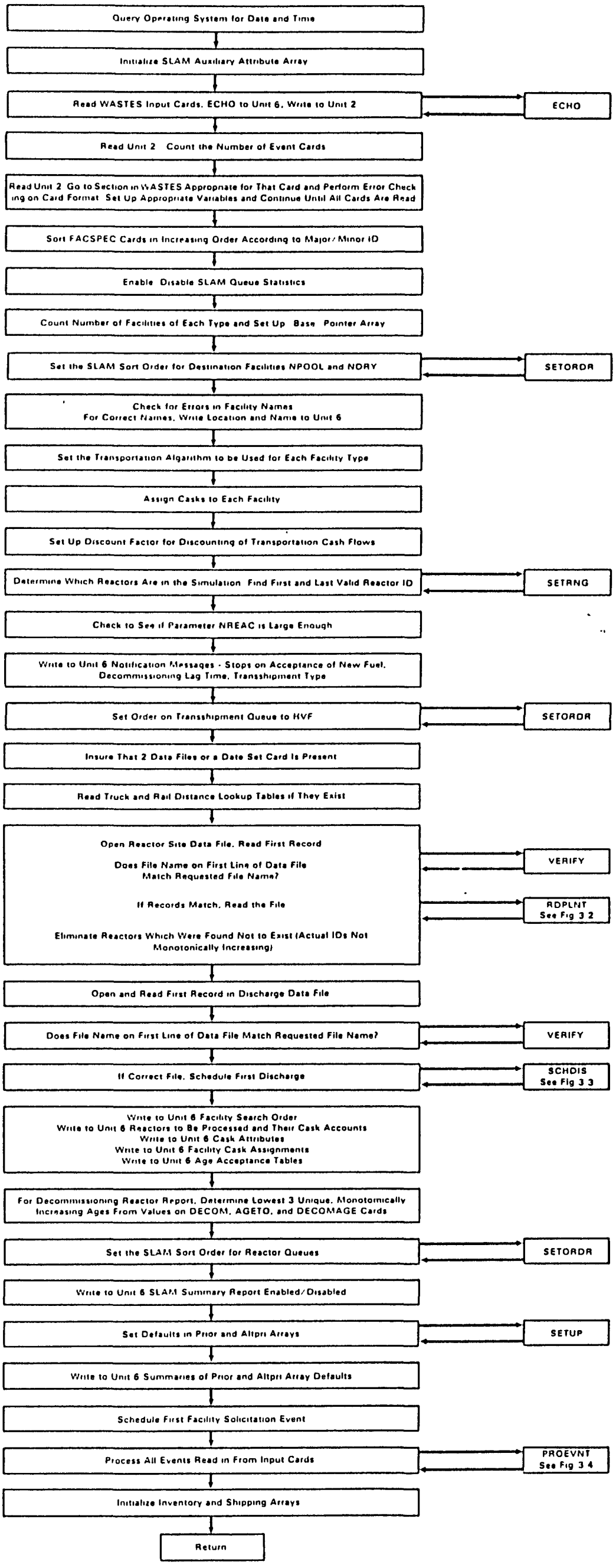


FIGURE 3.1. Block Flow Diagram of Subroutine INTLC

3.1 SUBROUTINE RDPLNT

Subroutine RDPLNT reads the entire reactor-site data file before the simulation begins (Figure 3.2). Reactors that have been excluded from the analysis are not read. Several arrays are loaded with the information required by WASTES to describe the reactor site.

A number of calculations relating to a reactor, if it is part of a shared pair, are performed in RDPLNT. If two reactors share a common pool, the available pool capacity equals the pool capacity minus the largest of the two FCR margins. A pool-shutdown event occurs when the latter pool is decommissioned. At the time of the first reactor's decommissioning, the FCR margin is adjusted if necessary. If two reactors share two pools, the available pool capacity equals the sum of the pool capacities minus the largest of the two FCR margins. Again, the pool shutdown event occurs at the time the latter pool's reactor is decommissioned.

RDPLNT also assigns reactors to the transshipment queue according to the option specified by the user. The way in which the transshipment network can be set up is discussed in Section 4.2.

If shipping/receiving casks have not been assigned to the reactor via the RXCASKS card, R1 and T1 casks are assumed to be available to reactors with rail capability, and T1 casks are assumed to be available to reactors with only truck access. The actions of the TMODE card can, however, override these default cask assignments. If a simulation is specified to have only rail access, all reactors will be assigned an R1 cask. If a simulation is specified to have only truck access, all reactors will be assigned a T1 cask.

3.2 SUBROUTINE SCHDIS

Subroutine SCHDIS reads in the data for the next reactor discharge batch and schedules a discharge event (Figure 3.3). Only the first discharge is scheduled before the simulation begins. During the simulation, the processing of a discharge event will trigger the scheduling at the next discharge event. In this way the number of events being maintained on the SLAM event calendar at any given time is minimized.

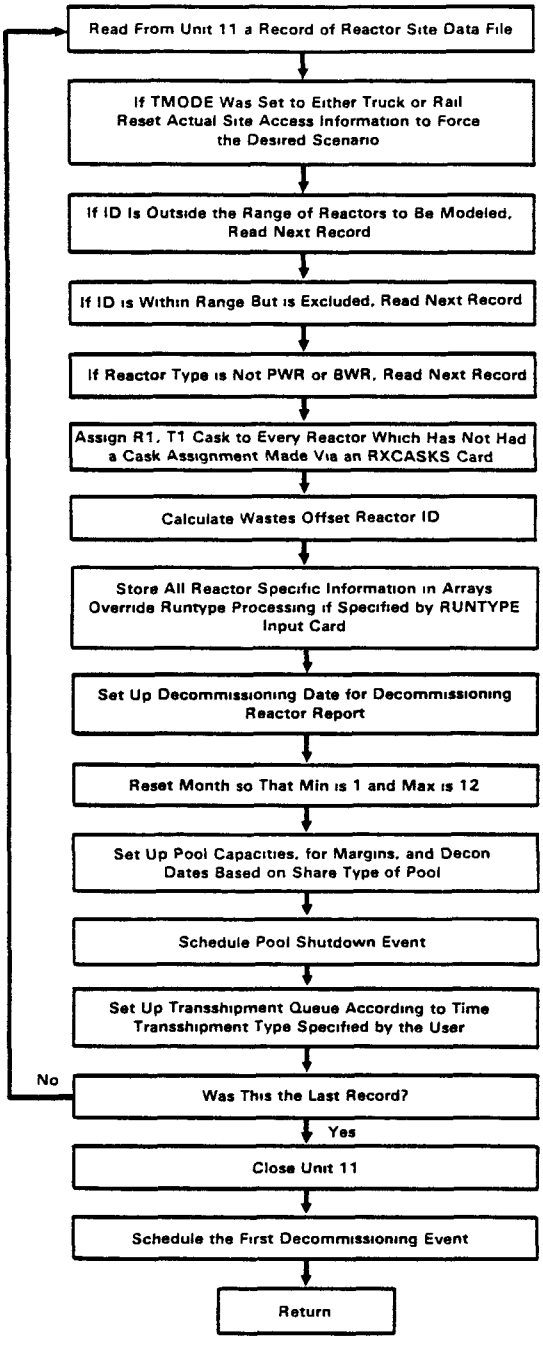


FIGURE 3.2. Block Flow Diagram of Subroutine RDPLNT

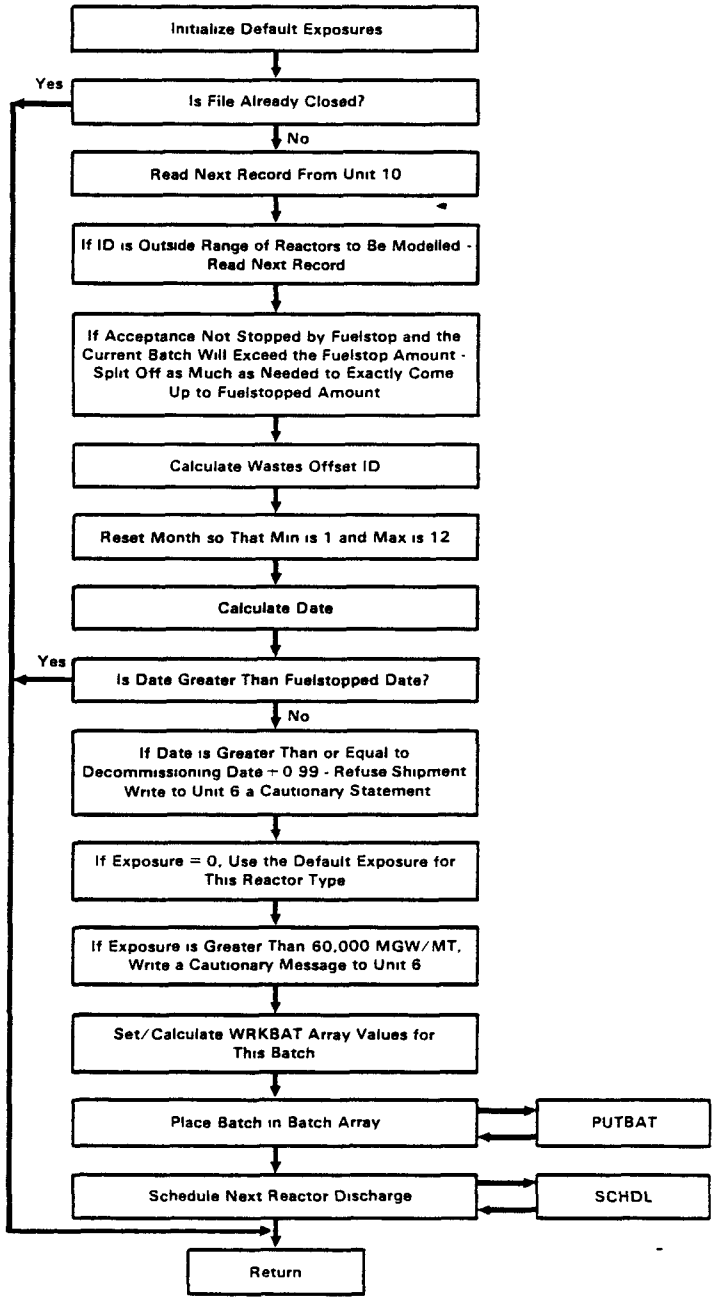


FIGURE 3.3. Block Flow Diagram of Subroutine SCHDIS

SCHDIS will not accept discharges from reactors not included in the simulation being run and will stop the acceptance of new fuel into the simulation at the appropriate point, if a FUELSTOP card has been included in the input stream. If the user has specified a date to stop accepting fuel, SCHDIS will refuse all fuel when the simulated time is greater than the time indicated on the FUELSTOP card. If the FUELSTOP card is based on MTU, SCHDIS will determine if the current batch will overflow the FUELSTOP amount indicated on FUELSTOP. The batch that would have exceeded the FUELSTOP amount is split to the amount that will just meet the FUELSTOP amount, and no more discharge batches are accepted.

If the fuel discharge has an exposure of zero when read in from the discharge data file, a default exposure of 33 MWd/kg for PWRs or 28 MWd/kg for BWRs is assumed.

3.3 SUBROUTINE PROEVNT

Subroutine PROEVNT schedules all WASTES input-card options that are events (Figure 3.4). The input cards that are processed by PROEVNT are CAPACITY, PRIOR, ALTPRI, DISCHARGE, TSHIP, and FCRFRAC. The subroutines that PROEVNT calls to set up these events are detailed in Table 2.1.

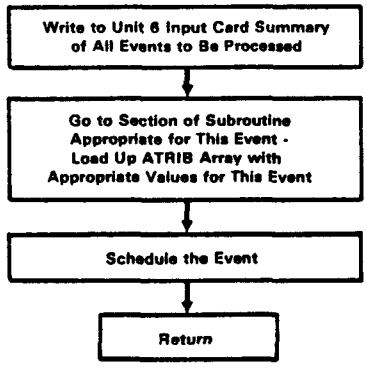


FIGURE 3.4. Block Flow Diagram of Subroutine PROEVNT

4.0 FORCED-DISCHARGE PROCESSING--SUBROUTINE RXDIS

The subroutine that processes reactor discharge events is RXDIS (Figure 4.1). After placing the spent-fuel discharge in the reactor's storage pool, RXDIS determines if this discharge violates the FCR storage margin for the pool (or shared pair of pools).

If the FCR margin has been violated, RXDIS will first attempt to ship the oldest material in the pool to any facilities that may be open to receive fuel. If, for any reason, the fuel cannot be shipped to a facility (e.g., no facility open, no remaining facility capacity, fuel unacceptable, or no shipping cask link), RXDIS will then attempt to transship the material. Any remaining material that exceeds the FCR margin after an attempt to transship will be placed in dry storage.

When attempting to ship to open facilities, RXDIS will search through the facility types in default order (REPRO, REPOS, MRS, FIS) unless this order has been superseded by an order specified on the FILLORDER card. After a facility type with existing capacity has been located, RXDIS will determine the shipping algorithm in effect for this class of facility (proximal by default). RXDIS then calls DSBCH and if all conditions are met, the shipment is initiated by DSBCH. Five conditions must be met to initiate transfer from a reactor to a storage facility: 1) the fuel in the reactor's pool must have been in the pool the required time, 2) the fuel in the reactor's pool must be acceptable to the facility, 3) the facility must have unused receipt capacity (both annual and total), 4) a shipping cask link must exist between the reactor and the destination facility, and 5) the PRIOR and ALTPRI arrays must be set to allow the destination facility (destination facility type) to look at reactor pools.

If any of the above conditions are not met, the shipment will not occur. If no shipment occurs or an insufficient amount is shipped because the five conditions cannot be met for the required amount of fuel and if the user has specified that transshipments can occur at this time, RXDIS will attempt to transship to other reactor pools.

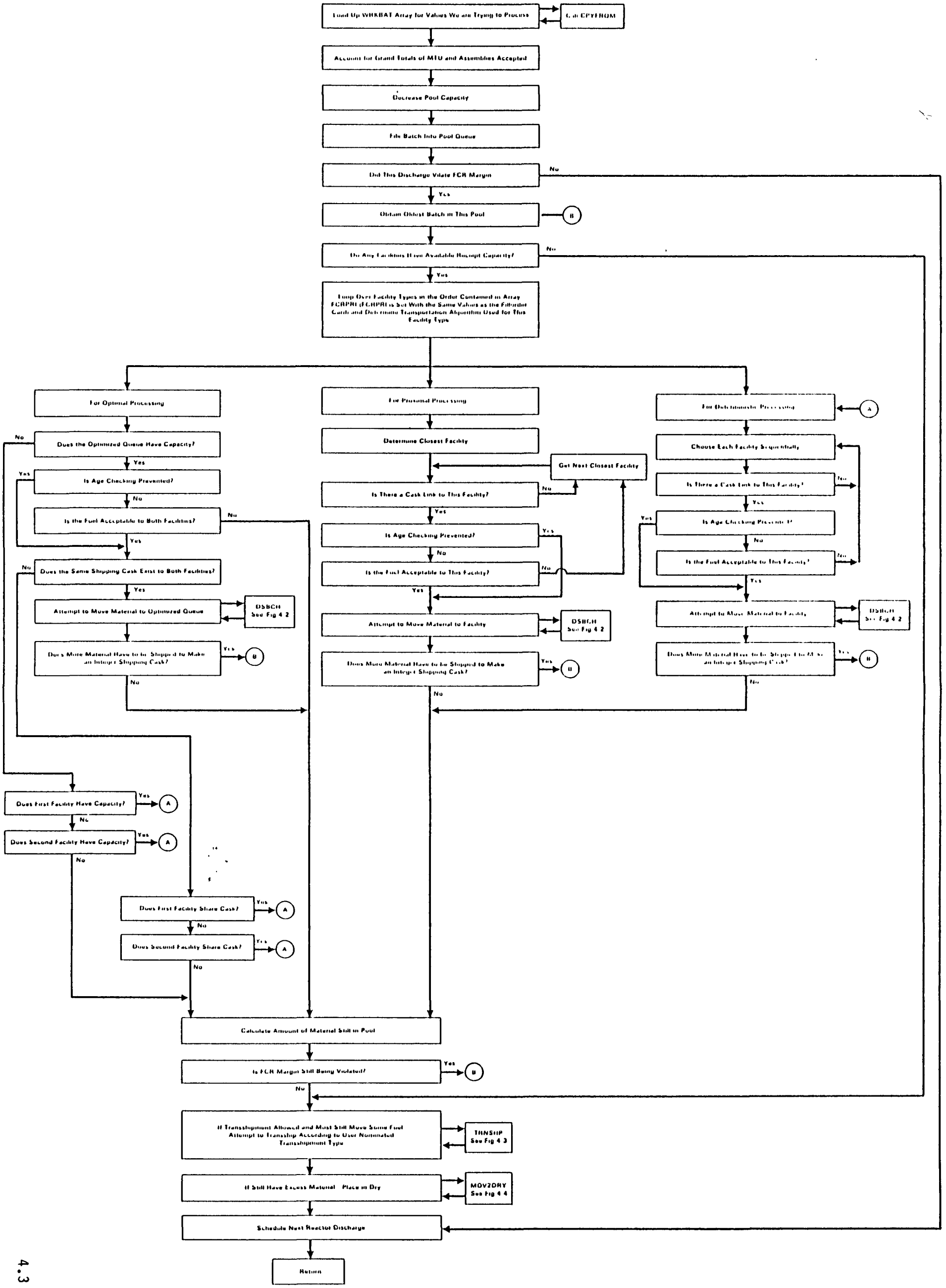


FIGURE 4.1. Block Flow Diagram of Subroutine RXDIS

If all of the preceding attempts to restore the FCR margin have failed, RXDIS moves whatever material is necessary into user-defined, at-reactor dry-storage casks (in full dry-storage cask quantities).

Since all of the actions that restore the FCR margin occur at the same simulated time as the discharge, the FCR margin of the simulated pool is never actually violated.

4.1 SUBROUTINE DSBCH

Subroutine DSBCH effects transfers of forced discharge material to the destination facilities (Figure 4.2). Spent fuel is immediately transferred to the destination facilities if they are being filled according to the proximal shipping algorithm or if it is acceptable to only one facility of the optimal pair. If the destination facility type is being filled according to the optimal shipping algorithm, the pointer(s) to the fuel batch(es) are removed from the pool queue and linked to the optimized holding queue for unloading at year-end.

DSBCH also returns a logical flag to RXDIS indicating whether or not an integer cask shipment has occurred. If the shipment was not an integer cask amount, RXDIS will continue to call DSBCH with a batch or sub-batch for shipment until the total transaction involves an integer number of shipping casks. Age constraints are ignored when filling out a cask to make an integer shipment.

4.2 SUBROUTINE TRNSHP

Subroutine TRNSHP transships material that exceeds a specified pool's FCR margin to other pools in the transshipment network (Figure 4.3). The destination pool for the transshipment is the pool that has the largest remaining capacity available to the shipping reactor within the transshipment network.

If a destination pool has sufficient capacity to accept at least one shipping cask and the oldest material at the source pool has met the minimum residence requirement, the shipment is made. If the destination pool has

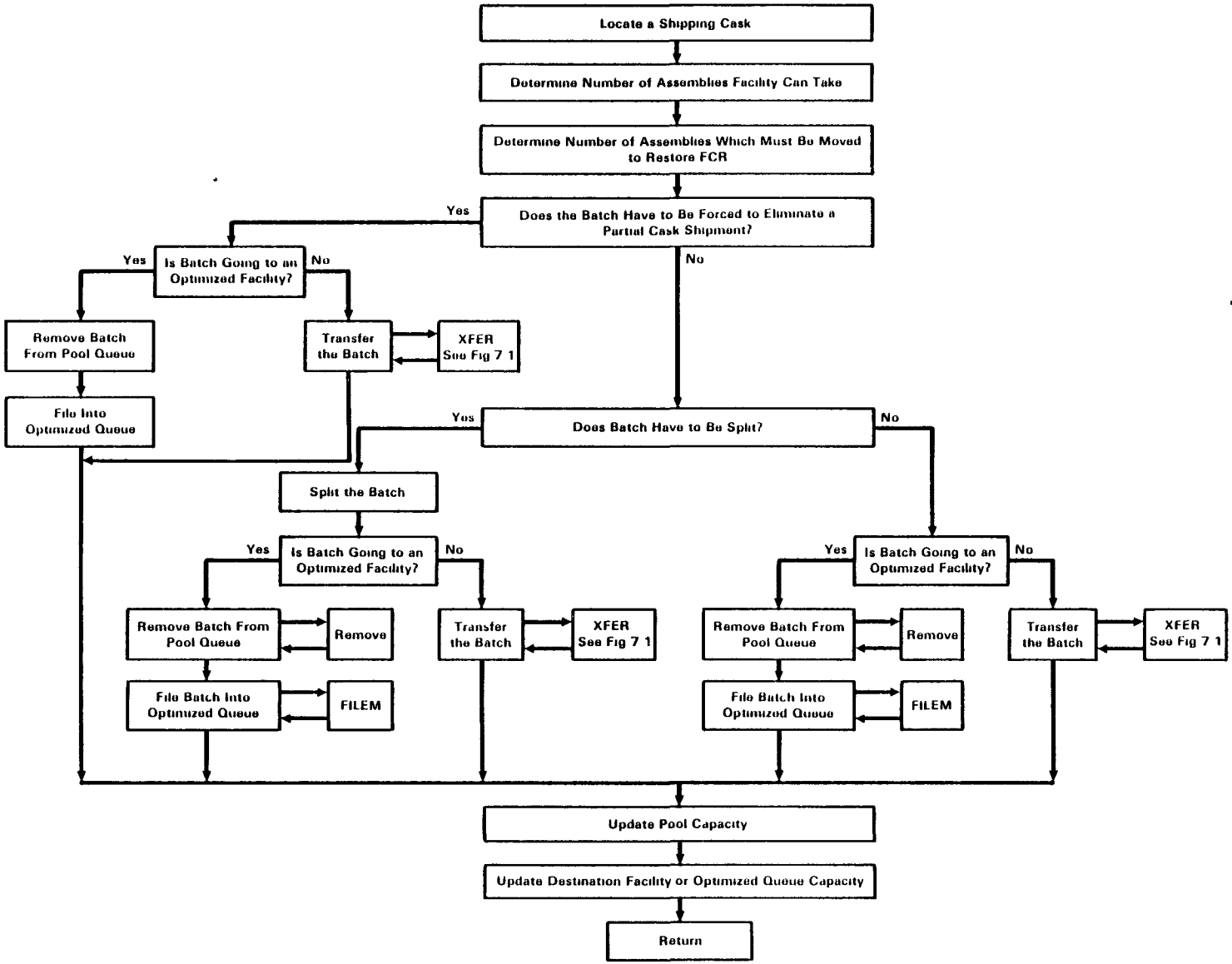


FIGURE 4.2. Block Flow Diagram of Subroutine DSBCH

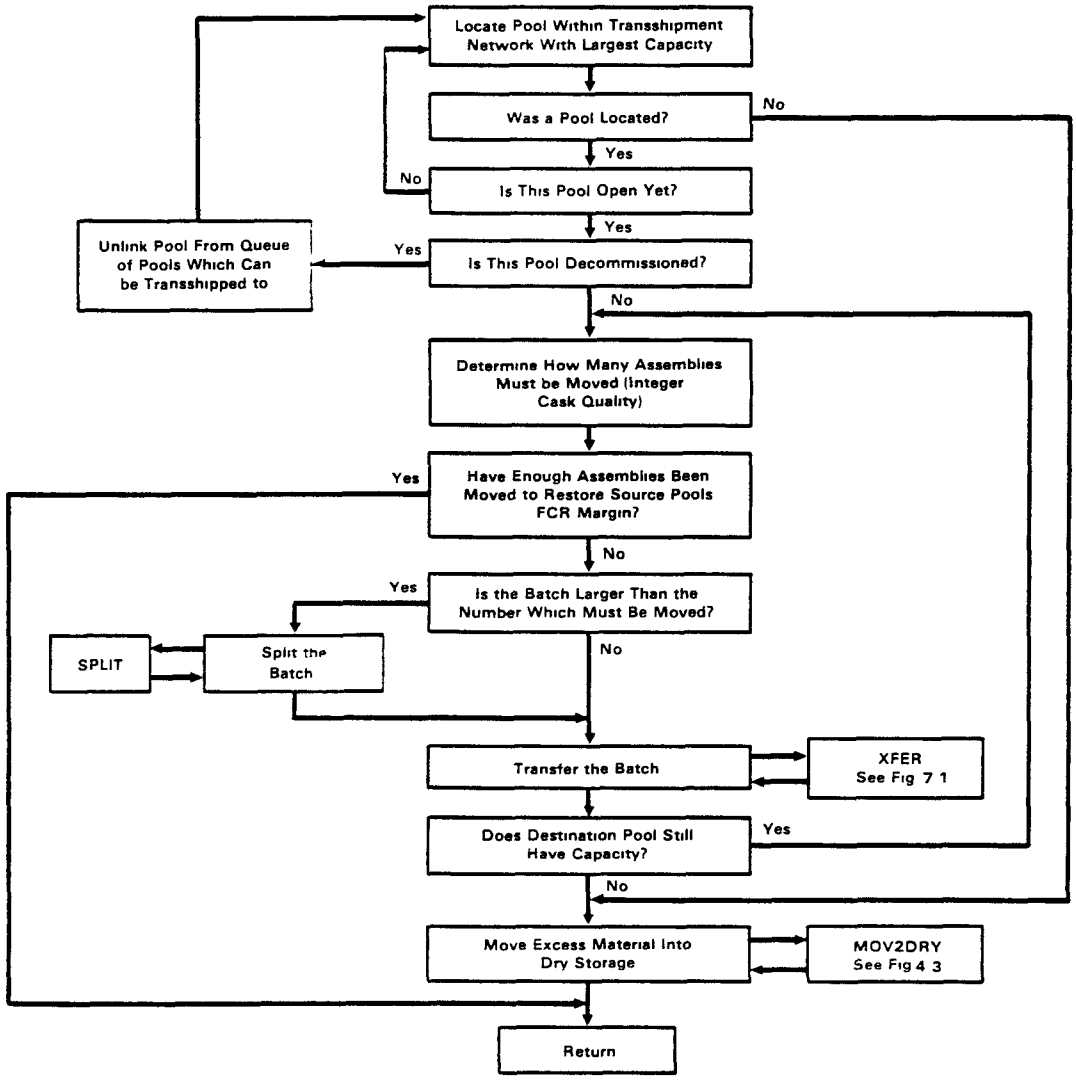


FIGURE 4.3. Block Flow Diagram of Subroutine TRNSHP

insufficient capacity to take all of the required shipment, transshipment to the next reactor within the transshipment group is attempted. After attempting to transship to all of the reactors within the transshipment group, any remaining assemblies that violate the FCR margin are moved to dry storage in integer cask quantities.

Transshipment can be user-specified to occur to storage pools of reactors that are the same type (i.e., PWR, BWR) that have opened and are not yet decommissioned. Transshipment to other pools is allowed if the reactors are 1) owned by the same utility, 2) located in the same state, or 3) specified by the user to be part of a transshipment network (this information is read from the reactor-site data file). If the reactor is placed in a transshipment network, transshipment to all other reactor pools in the same network is allowed.

4.3 SUBROUTINE MOV2DRY

Subroutine MOV2DRY moves excess fuel from the pool of a reactor to ex-pool dry-storage casks located at the reactor site (Figure 4.4). These dry-storage casks are filled in integer dry-storage cask amounts. Thus, the filling of a dry-storage cask will, in most cases, leave a small amount of capacity in the reactor's pool. The fuel that is moved into the dry-storage cask is the oldest fuel in the reactor's pool.

Since these fuel movements occur only within the reactor yard, only costs associated directly with dry storage are incurred. Shipping costs are not incurred.

The user specifies, via the DRYTYPE input card, the number of PWR and BWR assemblies that the cask can hold and the ex-pool storage technology that is being used. The technology being used is of primary importance for the dry-storage costing routines that are discussed in Section 8.3.

When a dry-storage cask is loaded, the amount of pool time available for cask loading operations is decremented 1.5 days by default.

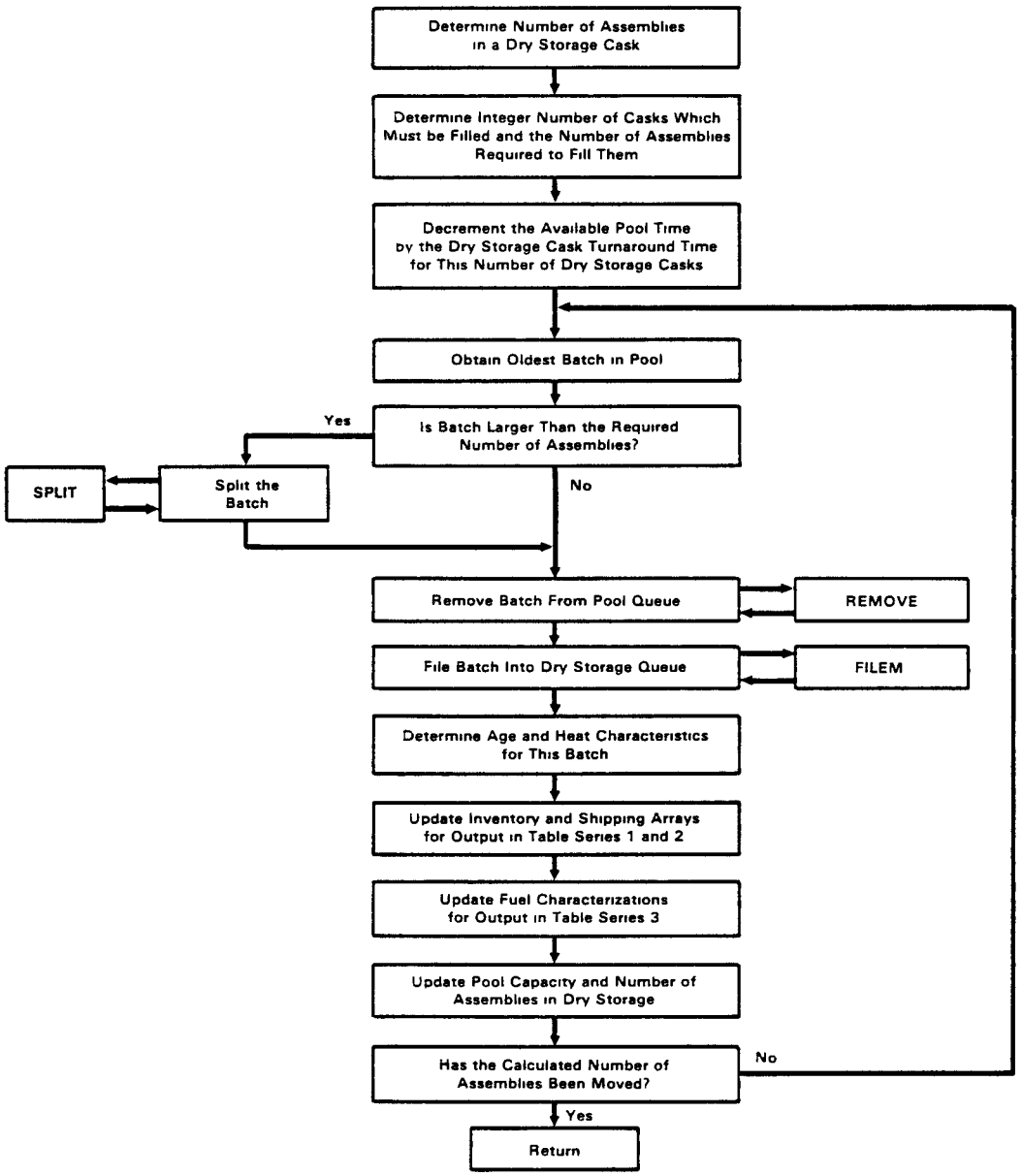


FIGURE 4.4. Block Flow Diagram of Subroutine MOV2DRY

5.0 DECOMMISSIONING REACTOR PROCESSING--SUBROUTINE DECOM

Subroutine DECOM moves fuel from reactors that are being decommissioned to destination facilities in the waste management system (Figure 5.1). The reactors processed by subroutine DECOM are given priority over normal facility solicitations (subroutine FAC) because the decommissioning event occurs earlier in each year being simulated. Additionally, DECOM attempts to unload all fuel at each individual reactor rather than selecting the oldest fuel from all reactors that are past their decommissioning priority date.

To maintain the sequence in which reactors will be decommissioned, a SLAM queue is used to sort the date upon which decommissioning will begin. The date that reactors are given decommissioning priority is determined by the reactor shutdown date from the reactor-site data file and the user-specified decommissioning lag time (DECOM card). If the sum of each reactor's shutdown date plus the lag time is within the time period being simulated, the reactor is given decommissioning priority beginning in the shutdown year plus the decommissioning lag time. When the reactor has been completely decommissioned (all fuel has been removed from both pool and dry storage), it is removed from the queue. For more information on the queue used, see Appendix D.

During decommissioning, the dry storage at each reactor is emptied before the reactor's pool is emptied. Integer dry-storage casking and integer-shipping casks are maintained. The only exception that may occur is for the last cask of fuel from a reactor pool. If the reactor does not have in storage a number of assemblies that is an integer multiple of the shipping cask being used, a single non-integer shipment will occur.

Reactors that are to be decommissioned have an additional shipping constraint. By default, each reactor is allowed 300 days per year that can be used for cask-loading operations. In DECOM, the amount of pool time that is available for loading shipping casks is decremented by the cask turnaround time whenever a shipping cask is loaded. This constraint prevents reactors from being decommissioned in a single year and then processing a number of shipping casks that would actually require several years to handle.

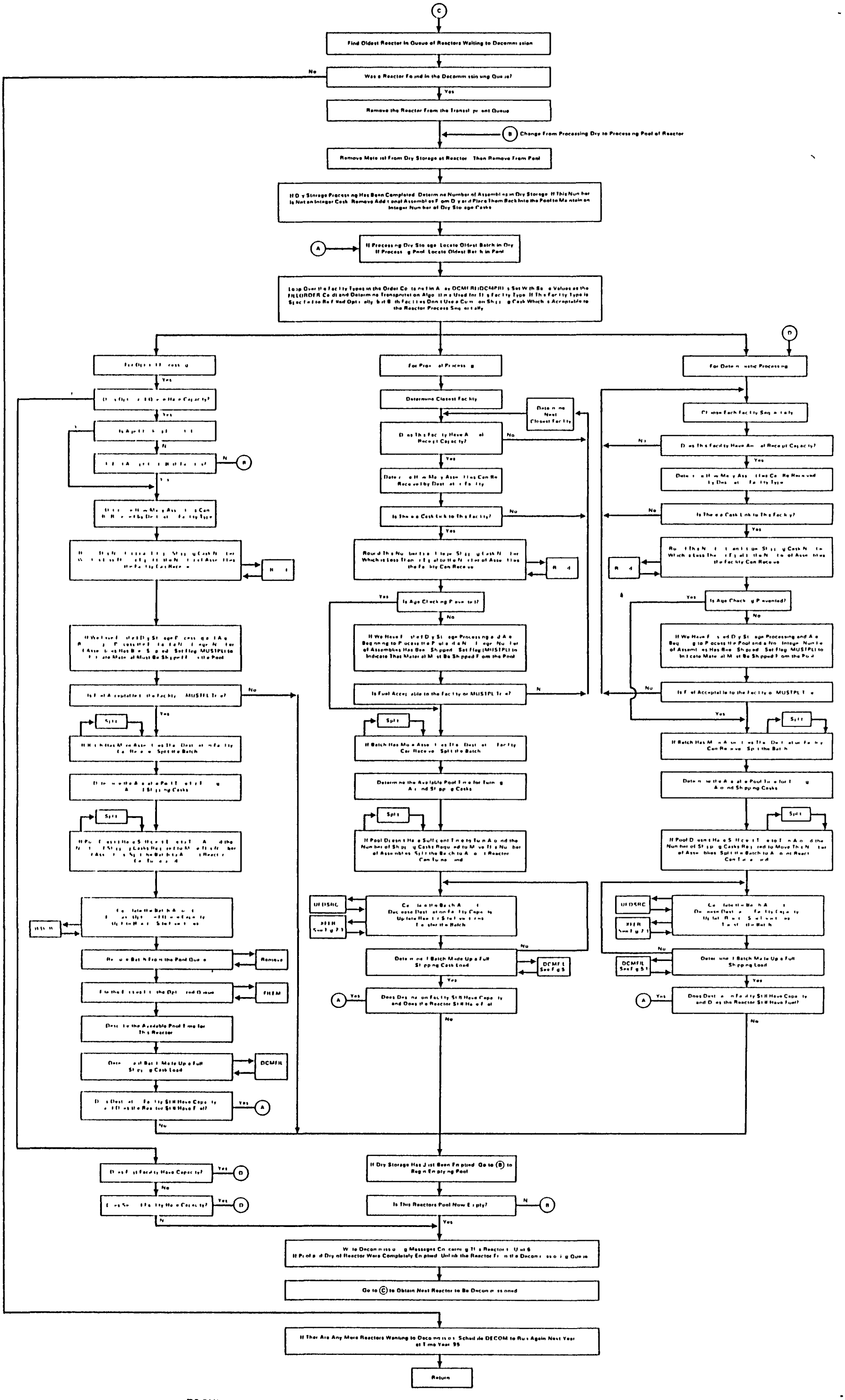


FIGURE 5.1. Block Flow Diagram of Subroutine DECOM

6.0 FACILITY SOLICITED FUEL TRANSFERS--SUBROUTINE FAC

Subroutine FAC attempts to fill the unused annual receipt capacity of all waste management facilities (Figure 6.1). It does this by calling the appropriate subroutine to process each class of facilities according to the proper shipping algorithm for that class. FRCAPO processes optimized transfers while FRCAPP processes fuel movements according to the proximal algorithm. FRCAPS is used when only one facility has unused spent-fuel receipt capacity.

All of the material transfers for the current year have occurred when FAC has completed all calls to FRCAPO, FRCAPP, and FRCAPS. At this time all fuel batch shipments that were routed according to the optimal shipping algorithm are held in the optimized work queue for each class of destination facility. BIOFMT is used to empty these work queues and actually makes the transfers from the holding queue to the destination facility and updates the inventory and shipping arrays.

Since FAC runs at the end of the year, a number of housekeeping operations are also carried out by this routine. The receipt capacities that each facility will have in the following year are reset and adjusted (up to 10 MTU) for any underrun or overrun experienced in the current year due to the shipment of full integer casks. If the amount of the overrun/underrun exceeds 10 MTU, the next year's annual capacity will be set to the nominal receipt capacity. For example, a repository may have a nominal annual receipt capacity of 3000 MTU for every year. If in a given year the repository receives 2995 MTU, the receipt capacity for the repository in the following year will be 3005 MTU.

To minimize memory requirements, all annual inventory and shipping information is written (by FAC) to a binary file that is read back while output is being processed at the end of the simulation. Annual information for the decommissioned reactor report (Table 1.01 of the WASTES output) is gathered when FAC calls UPDECOM. The annual shipping and inventory arrays are then reinitialized. Two error-checking routines can run concurrently with WASTES to ensure that dry storage is being maintained in integer cask amounts and

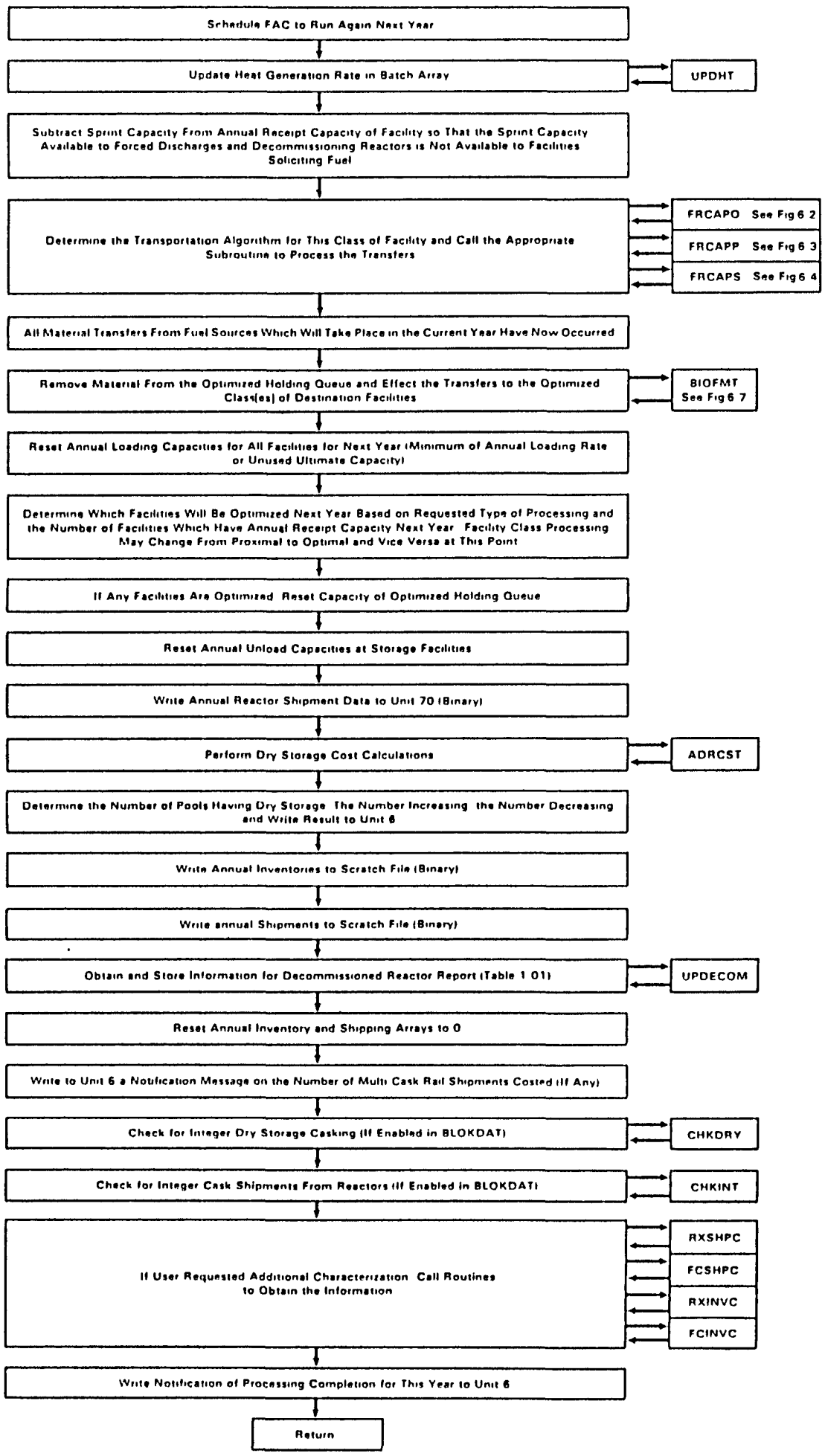


FIGURE 6.1. Block Flow Diagram of Subroutine FAC

that shipping casks are being filled in integer cask amounts. FAC calls these routines and performs these checks on an annual basis. If the user has requested additional characterization of shipments, four subroutines that gather this information are called.

6.1 SUBROUTINE FRCAPO

Subroutine FRCAPO fills the unused annual receipt capacity of classes of destination facilities so that the total shipping miles or the total shipping costs are minimized (Figure 6.2). In determining what fuel should be solicited in a given year, FRCAPO first processes the PRIOR loading priorities for each facility and then processes the alternate loading priorities for the class of facility. If either or both of the facilities in an optimized pair has been specified as the destination facility for processing by the PRIOR loading array, the destination facility(ies) looks individually and sequentially at the source facilities contained in the PRIOR array. These shipments are not routed optimally. Optimization occurs only when the alternate loading priorities are being processed and the destination facilities simultaneously look to facilities in the ALTPRI array.

In order for a shipment to be optimized, it must be acceptable to both facilities on the basis of age-acceptance requirements and the existence of a shipping-cask link from the source to both destination facilities.

If the shipment cannot be made optimally, an attempt is made to ship individually to each of the two possible destination facilities. If either of the possible destination facilities can accept the shipment, the shipment is made at once (fuel is not filed into the optimized queue).

As throughout WASTES, the spent fuel selected is the oldest material that can be located at the sources and that can be shipped to this class of destination facility. Also, optimized classes of facilities cannot receive multi-cask rail shipments.

6.2 SUBROUTINE FRCAPP

Subroutine FRCAPP fills the unused annual receipt capacity of classes of destination facilities so that the closest facility to the source receives the

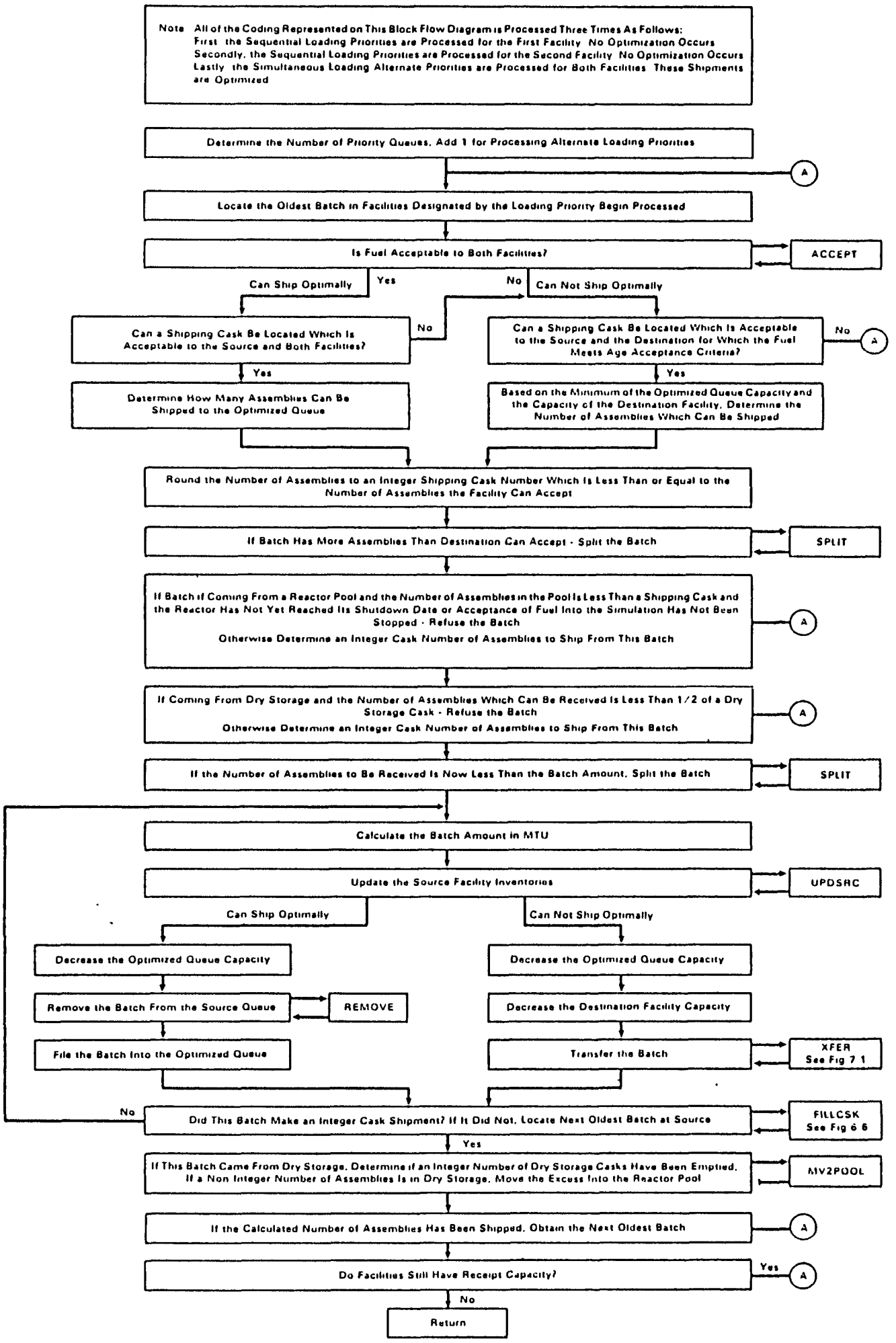


FIGURE 6.2. Block Flow Diagram of Subroutine FRCAPO

shipment (Figure 6.3). As each batch of spent fuel is located, FACORDR is called to locate the facility in the destination class that is closest to the source.

Facilities are first loaded individually and sequentially if they are the destination facility for PRIOR array processing priorities. After the PRIOR array priorities have been processed for each facility in the destination class, the ALTPRI priorities are processed. At this time, all facilities in the destination facility class simultaneously search source facilities in the ALTPRI array to locate the oldest fuel in the system.

If the source facility is an MRS facility and a multi-cask rail shipment card has been supplied that identifies the shipping cask used by this MRS facility, subroutine TRNLST is called and attempts to locate sufficient material at the MRS facility to build a multi-cask load for shipment by rail.

6.3 SUBROUTINE FRCAPS

Subroutine FRCAPS is used to fill the unused annual receipt capacity of classes of destination facilities when there is only one facility in the class or when a fuel batch is acceptable to only one facility of an optimized pair (Figure 6.4).

Facilities are first loaded individually and sequentially according to the loading priorities established in the PRIOR array. After the PRIOR array priorities have been processed for each facility in the destination class, the ALTPRI priorities are processed. Following this, each facility in the destination facility class simultaneously looks to source facilities in the ALTPRI array to locate the oldest fuel in the system.

If the source facility is an MRS facility and a multi-cask rail shipment card has been supplied that identifies the shipping cask used by this MRS facility, subroutine TRNLST attempts to locate sufficient material at the MRS facility to build a multi-cask load for shipment by rail.

6.4 SUBROUTINE TRNLST

Subroutine TRNLST attempts to obtain enough material from an MRS facility to make a multi-cask rail shipment and then makes the shipment (Figure 6.5).

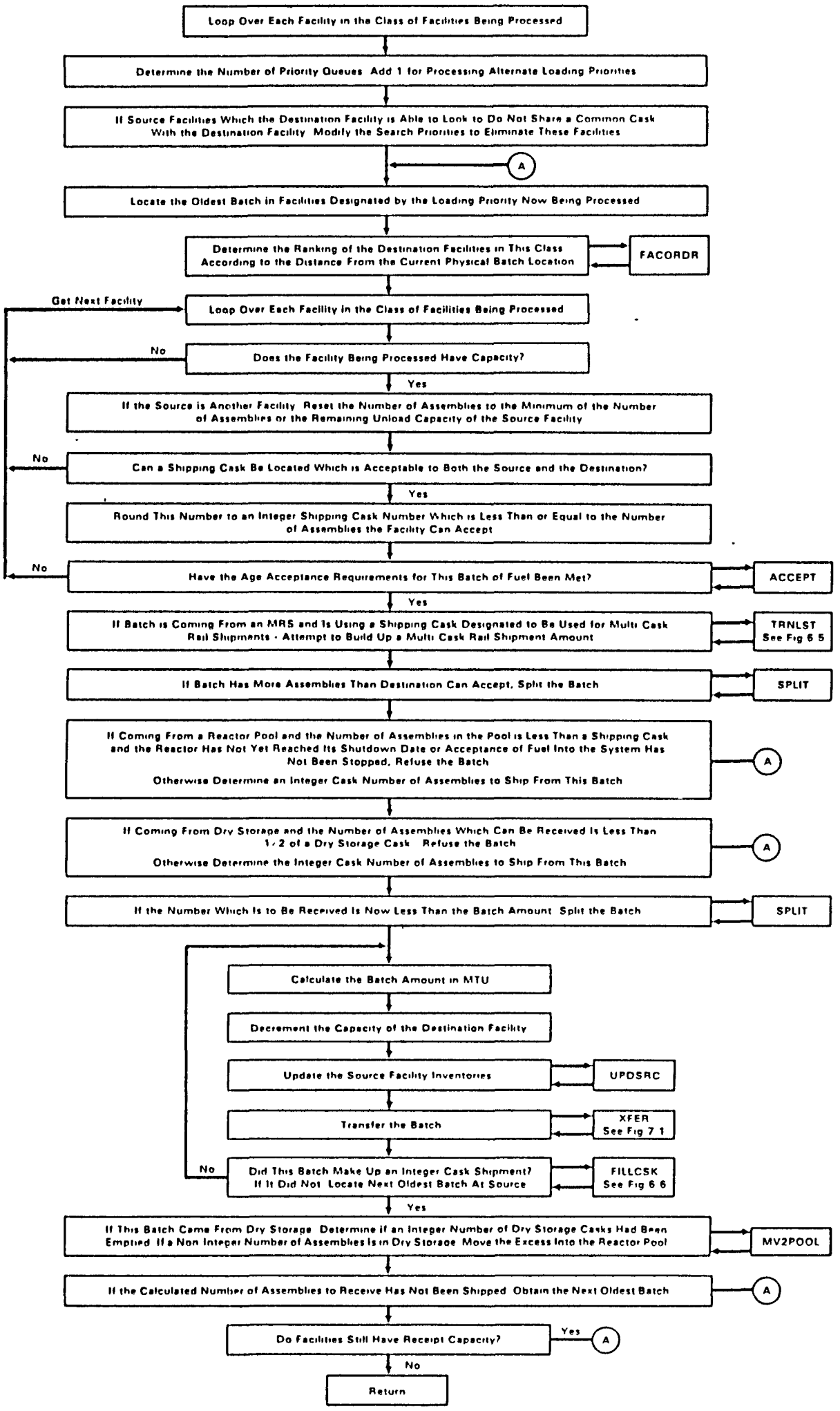


FIGURE 6.3. Block Flow Diagram of Subroutine FRCAPP

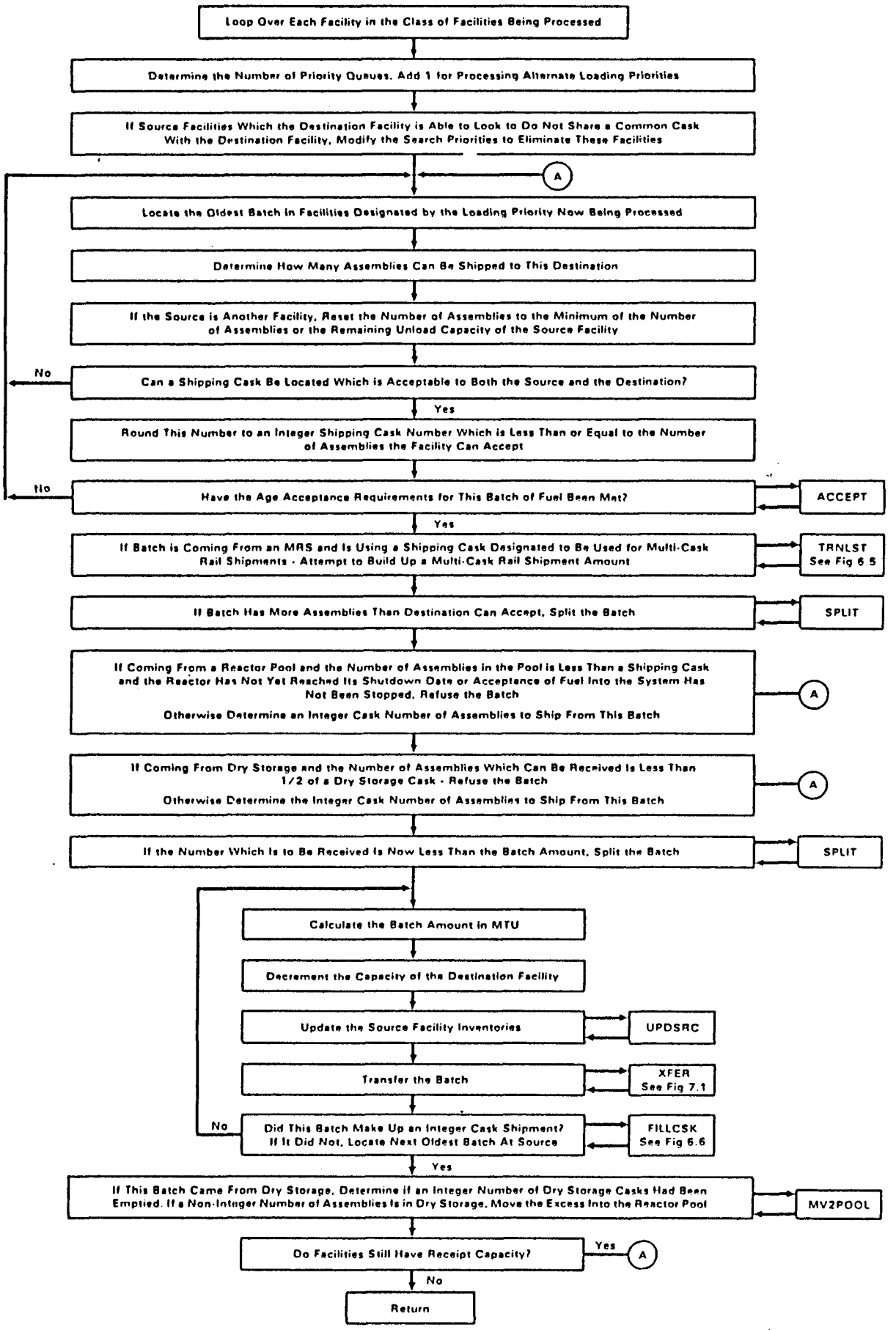


FIGURE 6.4. Block Flow Diagram of Subroutine FRCAPS

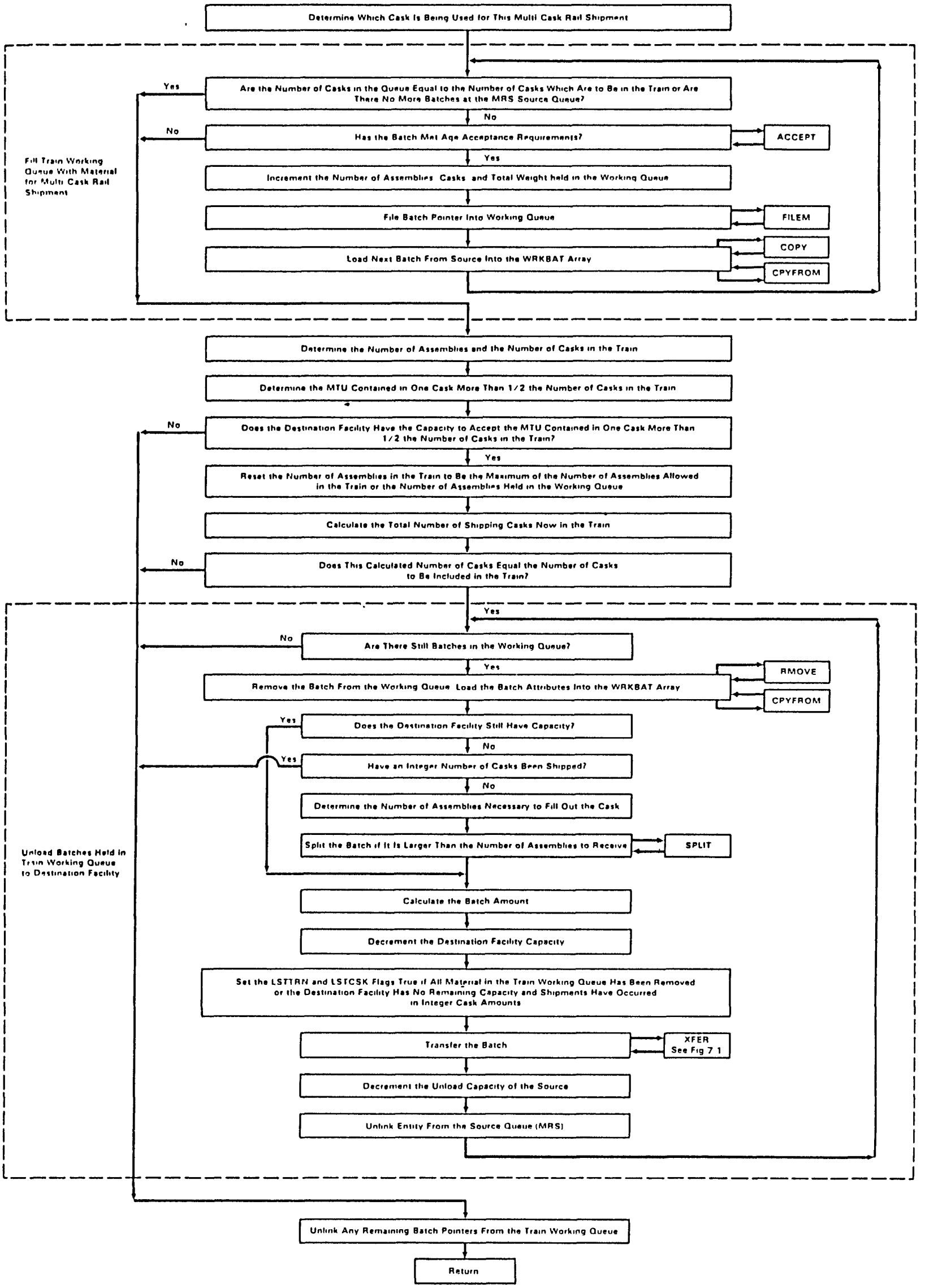


FIGURE 6.5. Block Flow Diagram of Subroutine TRANLST

TRNLST expects that if a multi-cask rail shipment is requested, the cask to be used for the shipment will be an unloading cask for the source MRS facility and a loading cask for the destination facility (typically a repository). TRNLST can be called only when the destination facility type is not processed according to the optimal shipping algorithm.

TRNLST first attempts to locate a sufficient amount of fuel at the MRS facility to fill all of the casks in the multi-cask shipment. The pointers to the fuel batches that may be included in the rail shipment (if enough fuel is located) are filed into a SLAM queue, which is used as a working queue. For more information on the working queue used, see Appendix D. Batch pointers are filed into the queue so that an integer number of casks is always maintained. PWR fuel and BWR fuel are not mixed within the same cask; however, trains hauling full casks of PWR fuel and BWR fuel are allowed.

TRNLST then checks to see that the destination facility has at least the capacity to accept the MTU contained in one half the train plus one cask, and that it has located sufficient material to make a full train. If either of these conditions is not fulfilled, the batch pointers are unlinked from the working queue (no rail shipment occurs) and TRNLST returns to the calling routine.

If both conditions are fulfilled, however, TRNLST removes spent fuel from the holding queue until the queue is empty or the destination facility runs out of capacity. If the destination facility runs out of capacity, TRNLST forces through sufficient material to fill the last cask so that shipping in integer cask amounts is maintained.

If large multiple cask shipments or large cask capacities are specified by the user, a significant overrun or underrun of the annual capacity can occur. This situation is discussed in Section 6.0.

When the last cask in a train is shipped, (either full or partially filled) the LSTTRN flag is set true. This flag indicates to the WASTES costing routines that the costs applicable only to multi-cask rail shipments should be applied in addition to regular transportation costs (shipping).

6.5 SUBROUTINE FILLCSK

Subroutine FILLCSK (Figure 6.6) is called after each batch is transferred by one of the FAC routines (FRCAPO, FRCAPP, FRCAPS) or after a reactor is decommissioned. The function of FILLCSK is to determine if an integer shipment has been made, and if not, to load the WRKBAT array with the oldest batch of spent fuel from the same queue as the previous batch. Additionally, FILLCSK sets a logical flag (FLAG) that forces the calling routine to transfer the batch immediately. If the first batch located by FILLCSK is larger than that required to fill the cask, the batch is split.

6.6 SUBROUTINE BIOFMT

Subroutine BIOFMT empties the optimized holding queue of each class of facility that was optimized after all transfers into the queue have been made (Figure 6.7). See Appendix D for more information on which SLAM queue is used to represent each of the classes of facilities. BIOFMT is called by FAC after all other material transfers in a year have occurred.

BIOFMT goes through the entire optimized working queue and makes transfers first to the facility with the lowest minor ID (as specified on the input cards) and then to the facility with the highest minor ID. Since the holding queue is sorted according to the difference in distances or costs, this will result

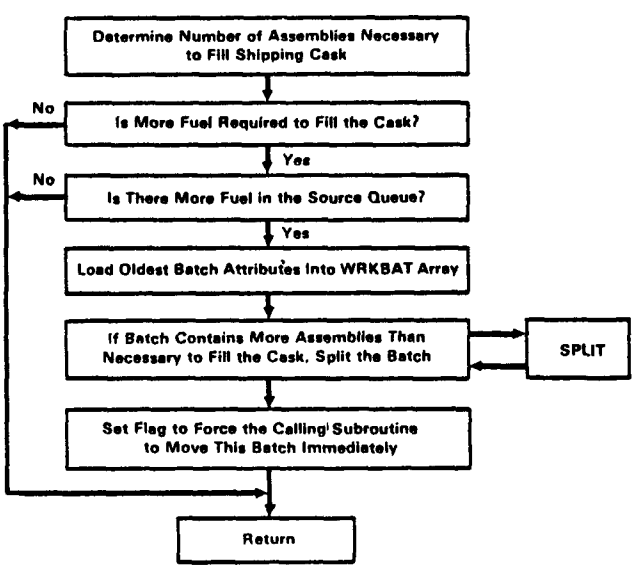


FIGURE 6.6. Block Flow Diagram of Subroutine FILLCSK

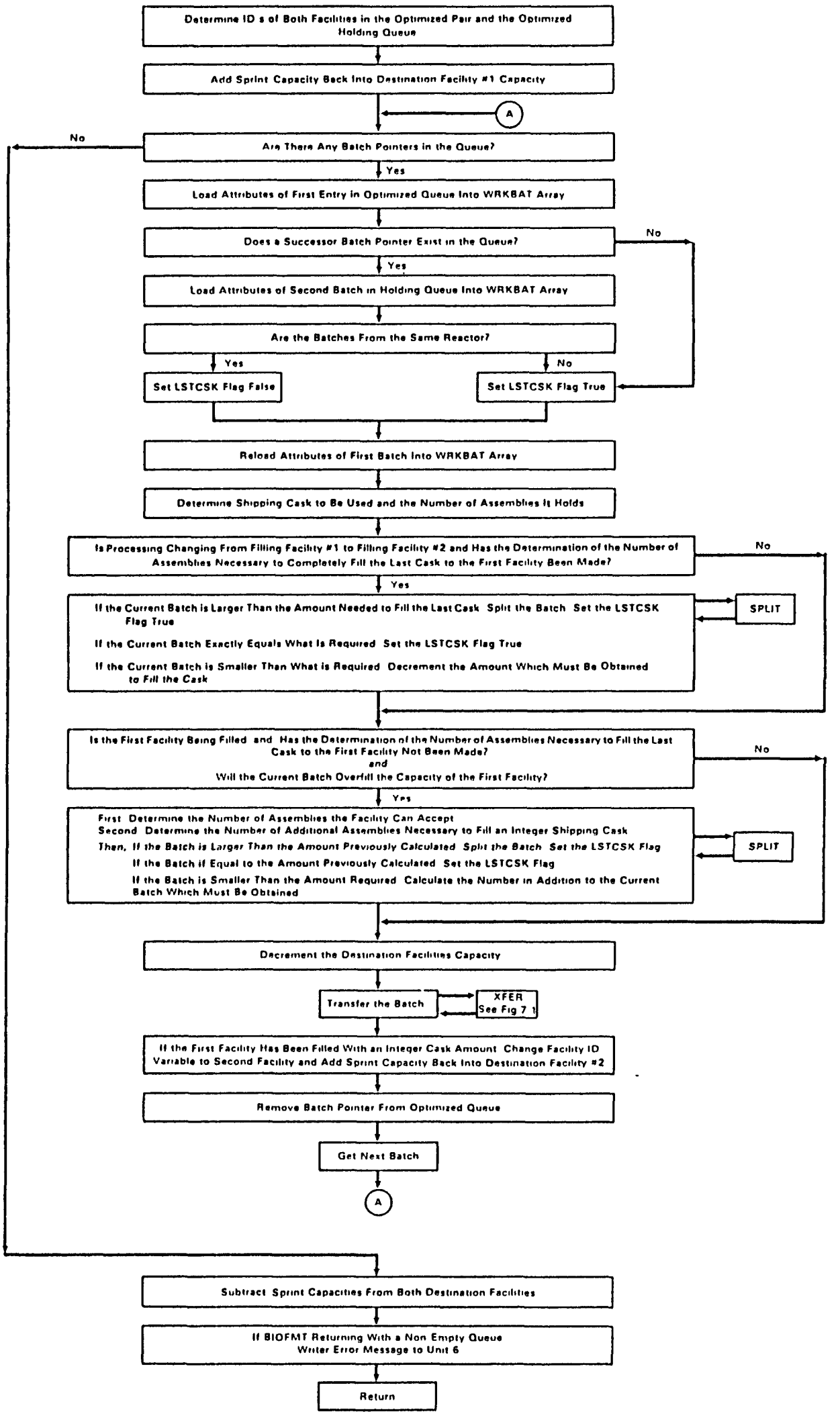


FIGURE 6.7. Block Flow Diagram of Subroutine BIOMFT

in an optimal routing of material under the assumption that the facility with the lowest minor ID will always attempt to fill its receipt capacity.

BIOFMT will always fill the first facility with an integer cask amount after detecting that the first facility has no remaining capacity. In this way the capacity of the first facility is overfilled each year by an amount less than one shipping cask.

7.0 TRANSPORTATION COSTING AND ACCOUNTING

This section reviews the WASTES subroutines responsible for making spent-fuel movements that incur transportation costs with the exception of fuel movements from reactor pools to at-reactor dry storage and from at-reactor dry storage back into the reactor pool.

These subroutines transfer a single batch of spent fuel at a time. In instances when a large shipment is occurring, such as shipping a multi-cask rail shipment or unloading the optimized holding queue, this set of subroutines is still called once for every batch that is transferred from the holding queues.

7.1 SUBROUTINE XFER

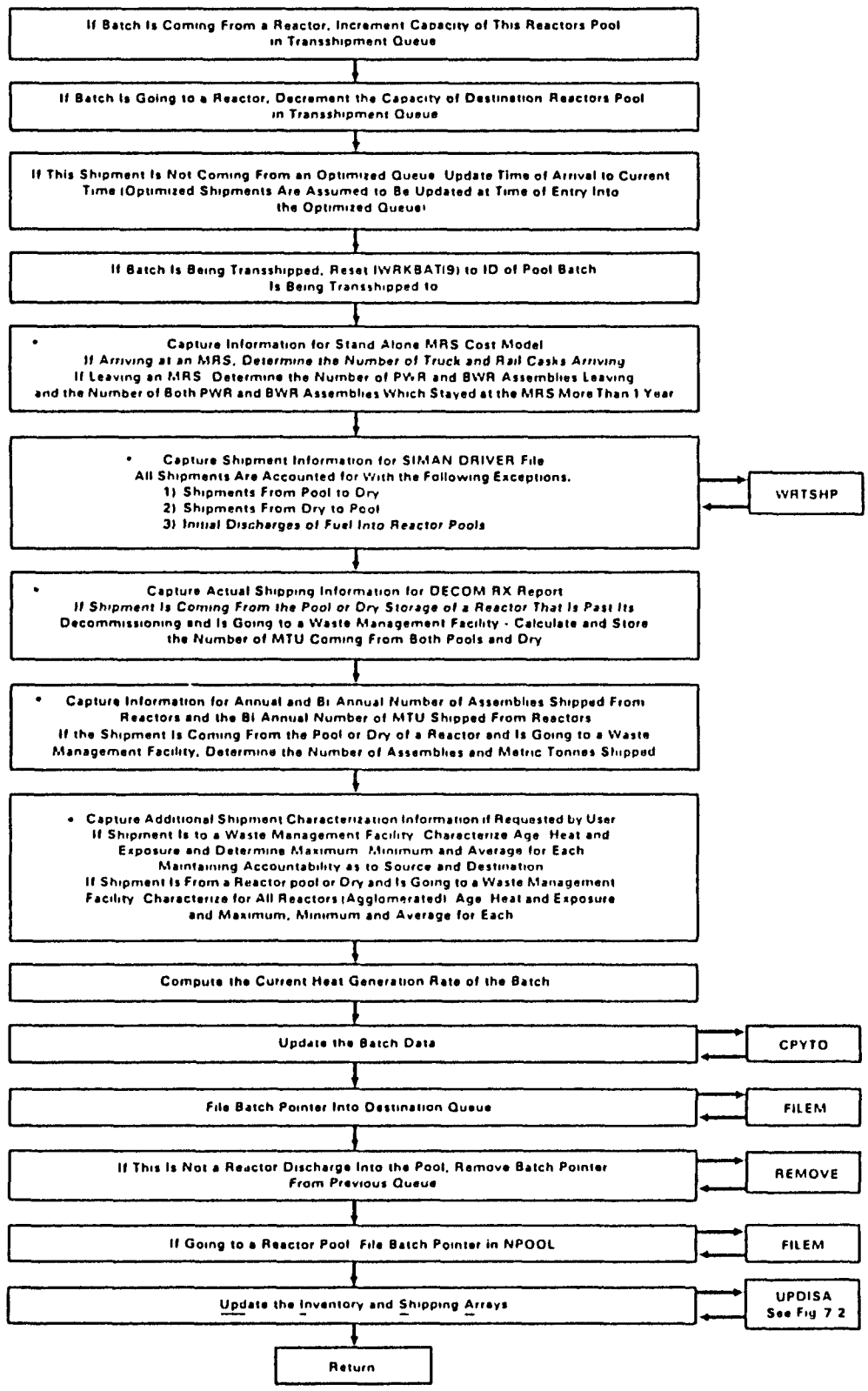
Subroutine XFER transfers individual batches of fuel from the source queue to the destination queue (Figure 7.1). Since every batch that incurs shipping costs is processed by XFER, the subroutine is also well suited for tracking or "capturing" information for a variety of outputs.

XFER captures information for the following special reports and output files: MRS cost model driver file, SIMAN dispatching model driver file, decommissioning reactor report (Table 1.01 of the output), biannual number of assemblies shipped by a reactor, biannual number of MTU shipped by a reactor, annual number of assemblies shipped by a reactor, and additional characterization information written to unit 1 (if user requested).

7.2 SUBROUTINE UPDISA

Subroutine UPDISA updates the inventory and shipping arrays for each batch as it is moved (Figure 7.2). These arrays contain inventory totals, receipt characterizations, shipping costs, number of cask busy days, number of cask miles, and reactor shipment statistics by centroid region.

As each batch is transferred, the number of shipping casks required or the percentage of the multi-cask rail shipment is determined. These percentages



*If Conditionals Inside These Blocks Are Not True, No Information is Captured

FIGURE 7.1. Block Flow Diagram of Subroutine XFER

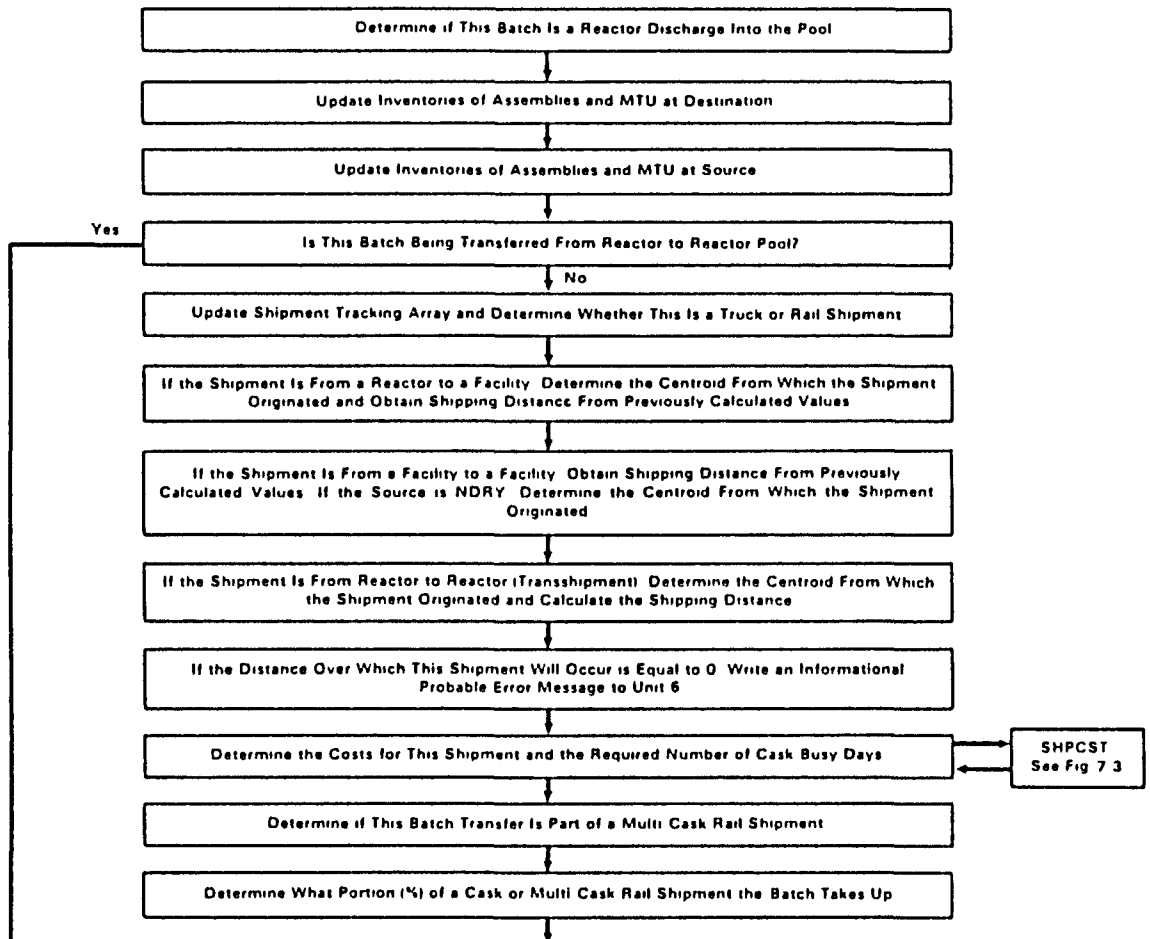


FIGURE 7.2. Block Flow Diagram of Subroutine UPDISA

are then applied to the various cask parameters to determine the cost of the shipment. For example, if a shipping cask contains 18 BWR assemblies and the current batch being transferred contains 6 BWR assemblies, 6/18 (or 33 percent) of the cost, busy days, and cask miles for shipping the entire cask will be calculated for this partial cask shipment. It is the responsibility of the calling routine to ship integer cask amounts.

The LSTCSK logical flag is used for those instances when an integer cask amount of fuel cannot be shipped (the very last cask from a reactor). Using the example from the previous paragraph with the LSTCSK flag true would result in the shipping costs being calculated for the weight of an empty shipping cask plus the weight of a cask 1/3 full. The cask miles, busy days, security and surcharges, and other applicable factors are calculated as if the cask is full (since they are equal).

The LSTTRN logical flag serves exactly the same function as the LSTCSK flag, however, it is set true once for each multi-cask rail shipment that occurs. This serves to apply those factors only applicable to the entire train shipment and to index the counter of the number of multi-cask rail shipments in the current year.

7.3 SUBROUTINE SHPCST

Subroutine SHPCST calculates the costs that would be incurred for shipping spent fuel by either legal-weight truck or general-freight rail modes of transport (Figure 7.3). The costs for safeguards and security charges are also estimated. In addition, the estimated cask transport time for each shipment is calculated.

The equations utilized in calculating these costs are based on the distance traveled and the weight incurred for each shipment. Separate equations are given for both truck and rail shipments for one-way distances less than and/or greater than 1000 miles. In addition, the equations provide an estimate of the safeguards and security charges that would be incurred for each shipment. All costs are reported in 1984 dollars. The equations used are shown below:

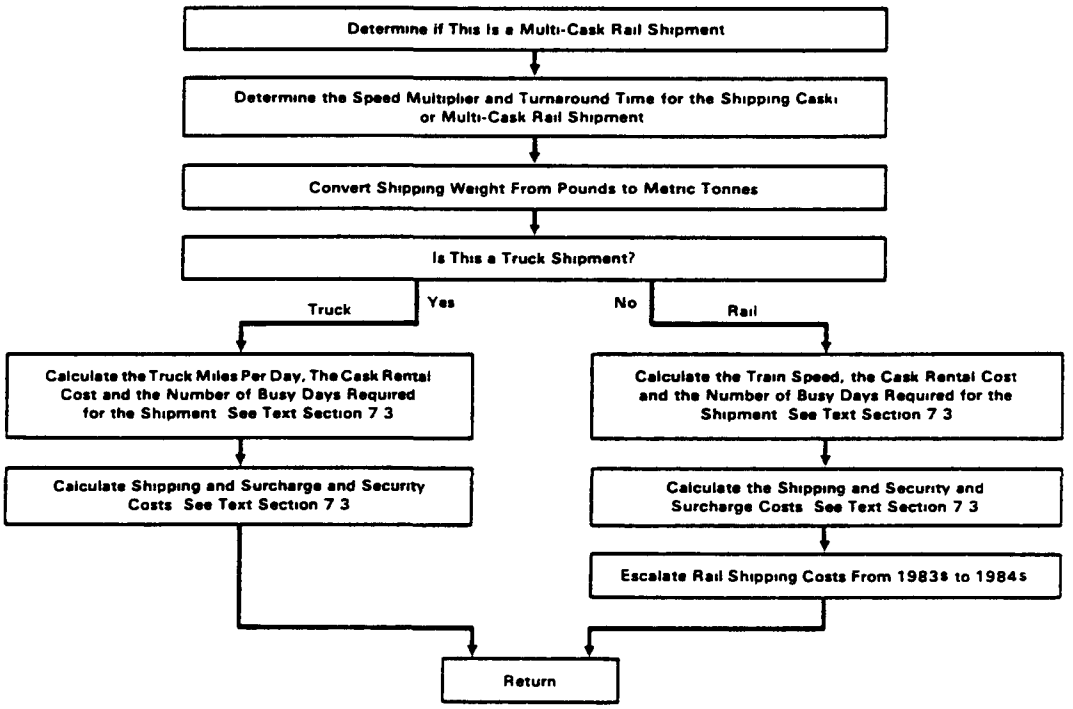


FIGURE 7.3. Block Flow Diagram of Subroutine SHPCST

Legal-Weight Truck

One-Way Mileage Less Than 1000 Miles

Shipping Cost ($) = [1.493 + .0033*DIS]*FWT
+ [0.428 + .0034*DIS]*EWT

One-Way Mileage Greater Than 1000 Miles

Shipping Cost ($) = [.0049*DIS - 0.16]*FWT
+ [.0040*DIS - 0.19]*EWT

For All Mileages

Safeguards/Security Cost ($) = [7.93*DIS(-0.1855)]*DIS

General-Freight Rail

One-Way Mileage Less Than 1000 Miles

Shipping Cost ($) = 1.053 * ([2.32 + 0.0067*DIS]*FWT
+ [2.15 + 0.0063*DIS]*EWT)

One-Way Mileage Greater Than 1000 Miles

Shipping Cost ($) = 1.053 * ([5.07 + 0.0040*DIS]*FWT
+ [4.72 + 0.0037*DIS]*EWT)

For All Mileages

Safeguards/Security Cost ($) = [291.65*DIS(-0.5987)]*DIS

where: DIS = One-way mileage
FWT = Loaded weight of cask (cwt)
EWT = Empty weight of cask (cwt)

The number of days that a cask will be utilized for each individual shipment is also estimated. The lease costs are then calculated by multiplying the total number of days that each cask is utilized by the daily lease charge for that cask type.

The number of cask-days utilized for a shipment is calculated as a function of the transit speed and the cask turnaround time.

Truck Speed = 840 miles/day
Rail Speed = maximum of 40 miles/day and (4.02 * DIS ** 0.48)

where: DIS = One-way mileage (miles)

The total number of cask-days utilized for a given shipment is calculated as shown below:

Cask-Days = [(2*DIS) / SPEED] + 2*TATIME

where: DIS = one-way mileage (miles)
SPEED = transit speed (miles/day)
TATIME = average turn-around time at each facility (days).

8.0 OUTPUT PROCESSING

Output processing in WASTES is accomplished by subroutines OTPUT, SPFOUT, LEVEL, and DRYCST/ADRCST. Thse subroutines are discussed in this section.

8.1 SUBROUTINE OTPUT

Subroutine OTPUT (Figure 8.1) is called by SLAM at the conclusion of the simulation to process the results. OTPUT writes information gathered during the simulation to various output files. OTPUT also generates the print file (unit 6) that is intended to be visually scanned by the user, and the post-processor data file (unit 1) that is intended to be read by post processors. Additional output requiring calculation is written by subroutines called from OTPUT.

Through PRXSHP, OTPUT writes the annual and biannual reactor shipment summaries to units 80, 95 and 96. OTPUT then writes a table (Table 1.0) of annual inventories for each facility in the simulation. The sum of all reactor pools and the sum of all dry storage are also written to unit 6 and a summary of this information is written to unit 1.

The decommissioned reactor report (Table 1.01) is then written to unit 6. The report conveys three distinct types of information, two of which are keyed according to age categories. The boundaries for the age categories are obtained from the lowest three, unique, non-zero, monotonically increasing ages read in on the DECOM, AGETO, and DECOMAGE input cards. However, if a DECLAG time of zero was supplied, the report is essentially turned off since in that case all of the age categories are set to zero. The report provides annual information on the number of shut-down reactors that still contain fuel. These reactors are categorized each year into one of the four age categories to summarize how many years past their decommissioning date they must maintain their spent-fuel pools. The second type of information characterizes the age of fuel onsite for all reactors that are past their date for decommissioning. The third type of information records shipments from the pools and dry storage of all reactors that have shut-down. In every case, the shut-down (or decommissioning) date is the date for the reactor that is read in from the reactor-site data file (in the case of shared reactor pools

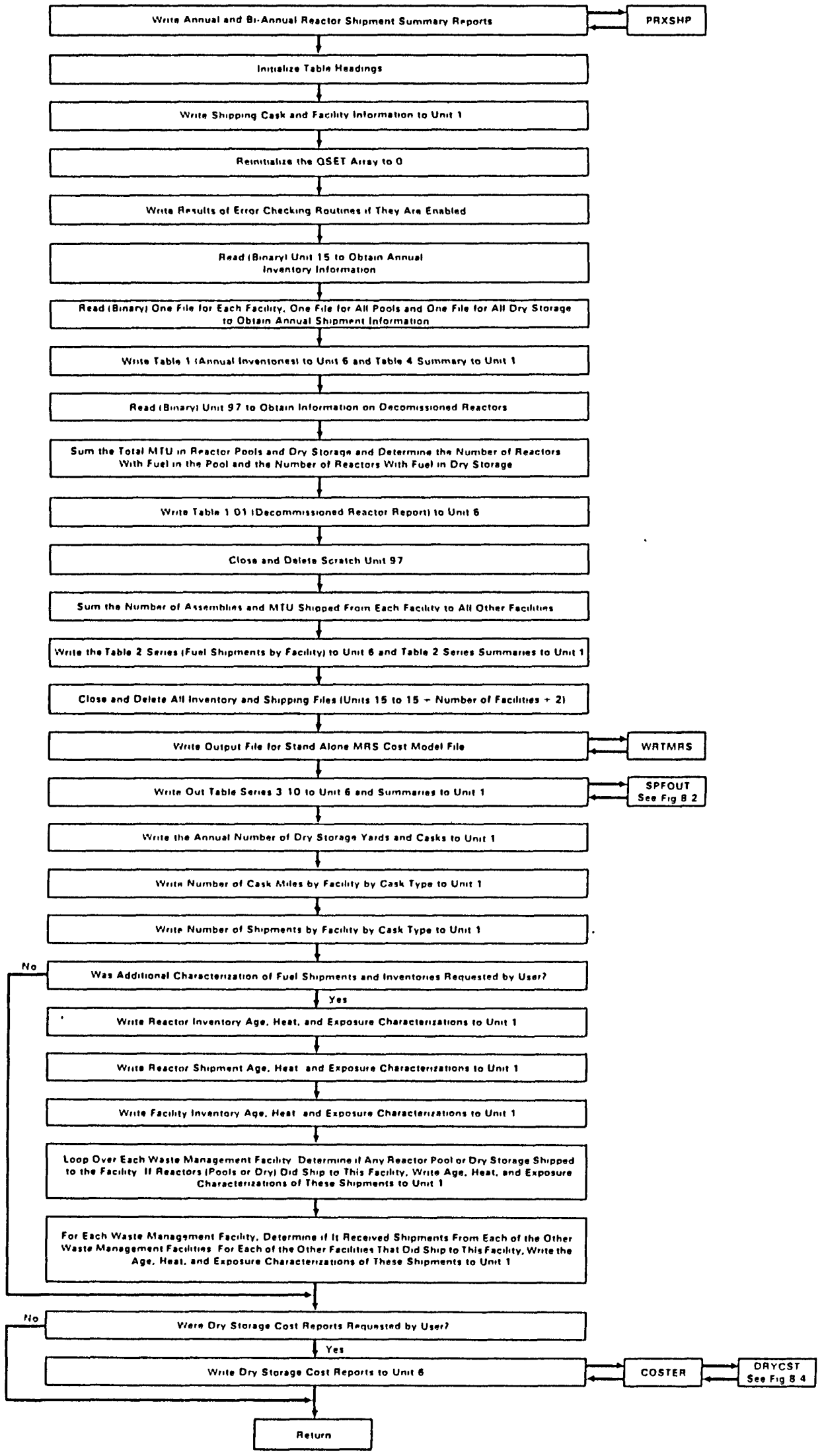


FIGURE 8.1. Block Flow Diagram of Subroutine OTPUT

when the decommissioning dates are not the same, this date is the latest of the two decommissioning dates).

Inventories of fuel shipped from each facility to every other facility (Table 2 series) are then written to unit 6, and a summary is written to unit 1. Control is then given to SPFOUT, which writes the Table 3 through 10 series of reports. Upon return from SPFOUT, OTPUT writes (if the user requests) additional characterization information to unit 1. Finally, if the user requests, dry storage cost reports are written to unit 6. Control is then returned to SLAM.

8.2 SUBROUTINE SPFOUT

Subroutine SPFOUT (Figure 8.2) writes the Table 3 through 10 series of reports to unit 6 and a summary of these reports to unit 1. Additionally, the levelized cask purchasing requirements are calculated by a call to LEVEL. The following section contains a brief review of the contents of each table series. A review of any assumptions that were made in the calculation of the tables is also included.

The Table 3 series segregates the MTU arriving at each facility into seven discrete fuel-age and seven discrete estimated heat-generation rate intervals. Additionally, the minimum, average, and maximum values are reported for both fuel-age and heat-generation rate annually. The average fuel exposure in each year is also reported. Average values presented in the Table 3 series are weighted averages based on the MTU in the batches that are accounted for.

The Table 4 series provides annual details on the characteristics of the transportation system arrivals at each facility. This information includes the total number of truck and rail shipments and the total metric tonnage shipped from each facility. The Table 4 series makes no independent assumptions.

The Table 5 series reports, by facility, the cost of shipping to that facility. Additionally, cask requirements (number of casks and cask busy days) are separated by truck and rail transportation modes and reported on an

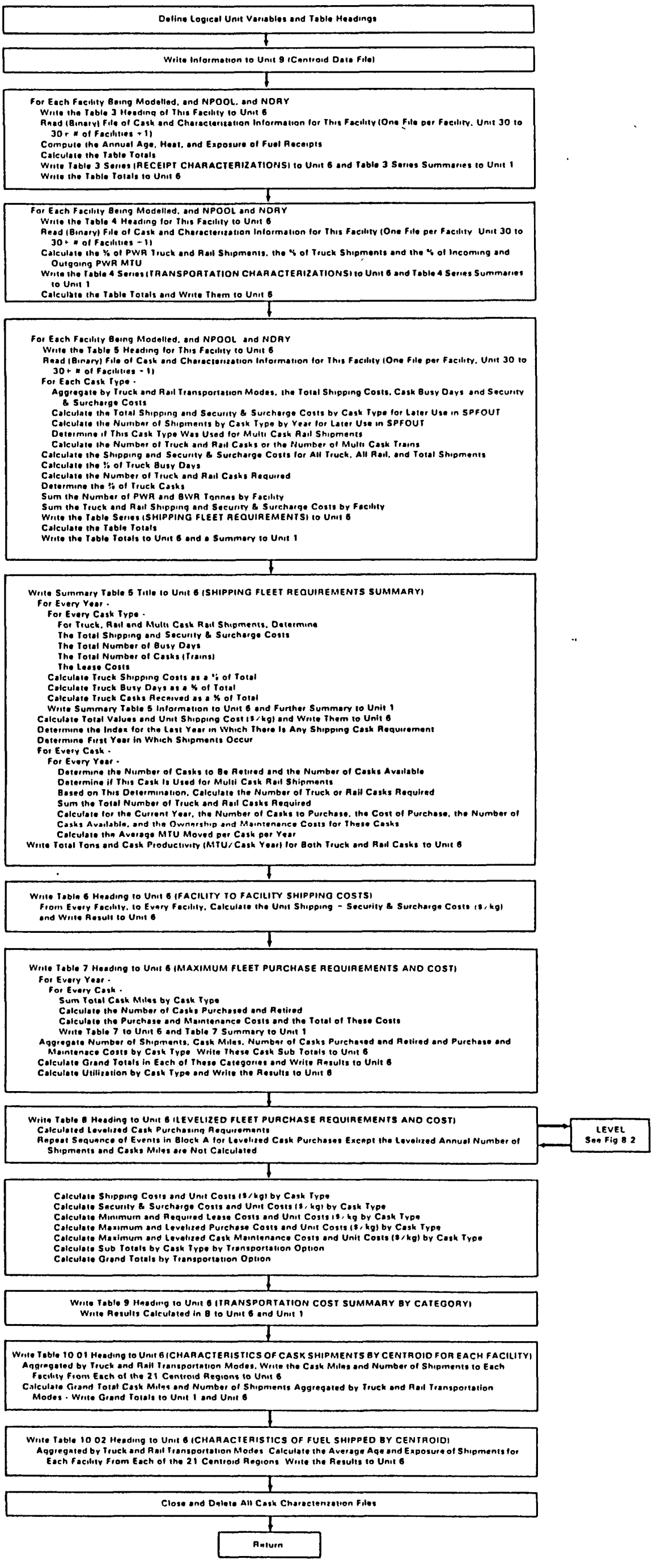


FIGURE 8.2. Block Flow Diagram of Subroutine SPFOUT

annual basis. In the Table 5 series, the costs reported under the heading SHIPPING COSTS are actually the sum of the shipping and the security and surcharge costs.

Table 6 summarizes the facility-to-facility shipping costs on a unit cost basis ($/kg). In Table 6, the unit costs reported are based on the sum of the shipping and the security and surcharge costs.

Tables 7 and 8, respectively, provide information on the maximum and levelized fleet purchase requirements. The number of shipments by cask type, the number of miles by cask type, and the number of casks available, purchased, and retired each year are identified. Estimates of the costs associated with purchasing and maintaining the cask fleets are also provided. In Table 7, the shipping cask utilizations expressed in units of cask days/year are calculated as follows:

$$\text{Cask Days/Year} = (\text{NNEED}/\text{NAVAL}) * \text{NDAYS}$$

where: NNEED = number of cask years of this cask type that are needed.
NAVAL = number of cask years of this cask type that are available.
NDAYS = number of days/year that this cask is available for shipping

The assumptions made in the generation of Table 8 are described in Section 8.2, which discusses the actions of the subroutine LEVEL.

Table 9 provides transportation costs broken down into the categories of shipping, security/surcharge, lease, purchase, and maintenance costs for four shipping scenarios. The shipping scenarios are: minimum lease, required lease, maximum purchase, and levelized purchase. Table 9 makes a number of implicit assumptions. Shipping costs and the security and surcharge costs are the same under each of the shipping scenarios. The method of calculating these costs is covered in Section 7.3. Lease costs are not applicable to the cask purchasing scenarios. The costs reported as minimum-lease costs assume that shipping casks can be leased for the exact number of cask days required. The costs reported as required lease costs assume that each required cask must be leased for the entire year. The formulas for calculating the lease costs are:

$$\underline{\text{Minimum Lease Cost}} = \text{CSKDYS} * \text{DLYRATE}$$

$$\underline{\text{Required Lease Cost}} = \text{ICSKREQ} * \text{DYSYR} * \text{DLYRATE}$$

where: CSKDYS = the number of cask days required in each calendar year

DLYRATE = the daily cask lease rate (user supplied on cask card)

ICSKREQ = integer number of casks required (e.g., if 5.63 casks are needed, ICSKREQ = 6)

DYSYR = the number of days/year the cask can be used (user supplied on cask card)

The costs reported as maximum purchase are generated on the basis of having as many casks available in each year as WASTES calculates are needed (casks available from previous years - casks retired + new purchases required). The costs reported as levelized purchase are calculated in accordance with the levelized cask purchasing algorithm described in Section 8.2. Ownership and maintenance (O&M) costs are not applicable to the cask purchasing scenarios. The O&M costs are the same for both cask leasing scenarios and are calculated as follows:

$$\text{O\&M\$} = [\ \text{NCASK} * \text{RATE}\ ] \text{ summed annually}$$

where: NCASK = number of casks of each type required in each year

RATE = annual lease rate for this cask type

These costs are summed annually for every year in which the cask exists, except that no O&M costs are calculated after the last year that had a cask requirement of a given type of shipping cask. Therefore, even if casks of one or more types are idle, they will still accumulate O&M costs until the end of the simulation or until there are no future requirements for any shipping cask of any type.

The Table 10 series characterizes shipments in user-specified aggregates (centroids). Table 10.01 reports details on the number of casks shipments by centroid, while Table 10.02 provides information on the average age and exposure as shipped from each centroid. No independent assumptions are made for the Table 10 reports.

8.3 SUBROUTINE LEVEL

Subroutine LEVEL (Figure 8.3) contains a levelized cask purchasing algorithm. This algorithm is a first order approximation of a cask purchase/replacement method in which the "peak requirements" in a given year may be transferred into an earlier year. The method discussed in this section acts as a "levelizer" by moving cask requirements from years that have peak requirements to prior years that had lower requirements. The cask levelizing algorithm does not affect the WASTES simulation, but is based on simulation results. It provides a useful tool for estimating cask requirements under a levelized cask purchasing scenario.

For each shipping cask type modeled in the simulation, the algorithm allows up to 300 MTU of lagpool storage (10 percent of the annual receipt rate of a waste management facility having a 3000 MTU annual receipt rate) to be available for levelizing peak year transfer requirements.

The algorithm examines the annual cask requirements that were previously calculated by WASTES and compares the number of casks required in a given year with the number of casks previously purchased. If the required number of casks exceeds the number of casks previously purchased, additional casks are purchased in that year. If the required number of casks is less than the number of casks available, the excess casks become available to remove spent fuel from future years in which the number of casks required will exceed the amount then available. All future years will be reduced until the excess number of casks in lagpool storage is completely eliminated.

The levelizing algorithms also ensure that a cask will be available only during its prescribed lifetime (user input). Once a cask has reached the end of its lifetime, it is either replaced (new purchase) or retired. The number of casks purchased and retired is based on the levelized cask requirements.

The initial lagpool and reduced lagpool amounts are expressed in cask years of average shipping cask productivity and are calculated according to the following formula:

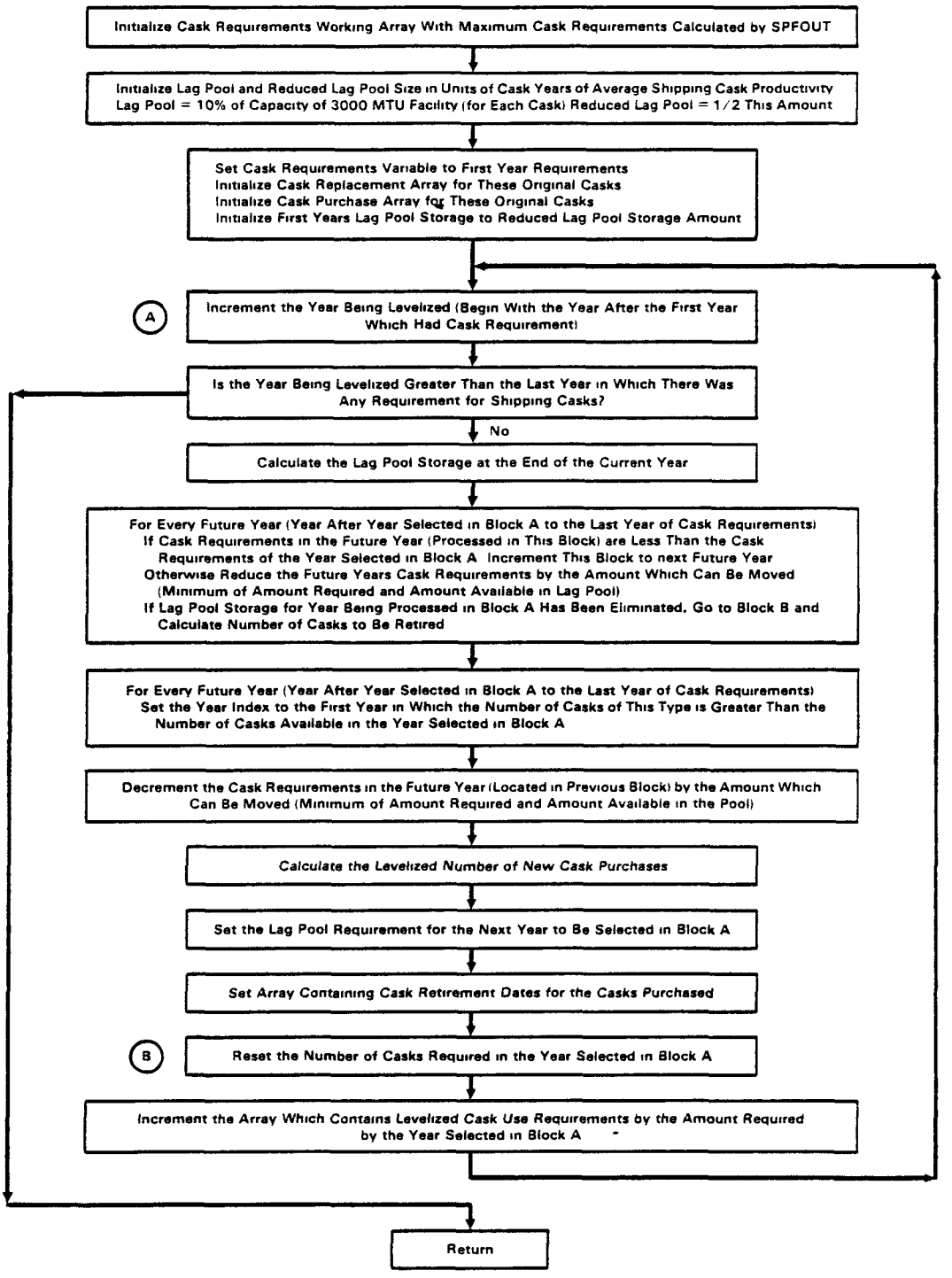


FIGURE 8.3. Block Flow Diagram of Subroutine LEVEL

$$LPMAX = IFIX(300. / AVMTPC + 0.999)$$

$$LPRED = LPMAX / 2$$

where: LPMAX = maximum size of lagpool expressed in units of shipping cask years of average cask productivity

LPRED = reduced lagpool size expressed in units of shipping cask years of average cask productivity

AVMTPC = the average cask productivity for the cask being levelized in units of MTU/year

300. = lagpool size in MTU

A description of the method used for calculating the amount of lagpool storage that may be moved forward into previous years follows. For each year in which there were shipping cask requirements, the lagpool size available for reduction of future years' cask requirements is:

$$ILAG = MIN(ICSK-NCREQ(NCASK,IYR), LPMAX-ILP(IYR-1))$$

$$ILP(IYR) = ILP(IYR-1) - LPRED + ILAG$$

$$ILP(IYR) = MAX(ILP(IYR), 0)$$

where: ILAG = lagpool size for the year being processed.

ICSK = number of cask years required

NCREQ = array containing the number of casks years required of this cask type in the year being processed (may have been partially levelized already).

ILP = array containing the size of the lagpool for each year.

MAX and MIN = intrinsic FORTRAN arithmetic functions that respectively obtain the maximum and minimum values of a variable pair

Based on this amount of lag pool storage available, the future year requirements are reduced as follows:

$$IREM = NCREQ(NCASK,K) - ICSK$$
$$IRED = MIN(NCREQ(IREM, ILAG)$$
$$NCREQ(NCASK,K) = NCREQ(NCASK,K) - IRED$$
$$ILAG = ILAG - IRED$$

where: IREM = the shortfall in the number of cask years required (number of cask years required - number of cask years available)

IRED = the number of cask years by that future years requirements can be reduced

8.4 SUBROUTINES DRYCST/ADRCST

Subroutine DRYCST (Figure 8.4) writes a dry storage cost summary for each of the dry storage cost categories used in the simulation and an aggregate summary for all categories. The dry storage cost categories are listed in Table 8.1.

DRYCST calculates the minimum, average, and maximum costs; the metric tonnes; and the unit costs for each dry storage costing category, and then

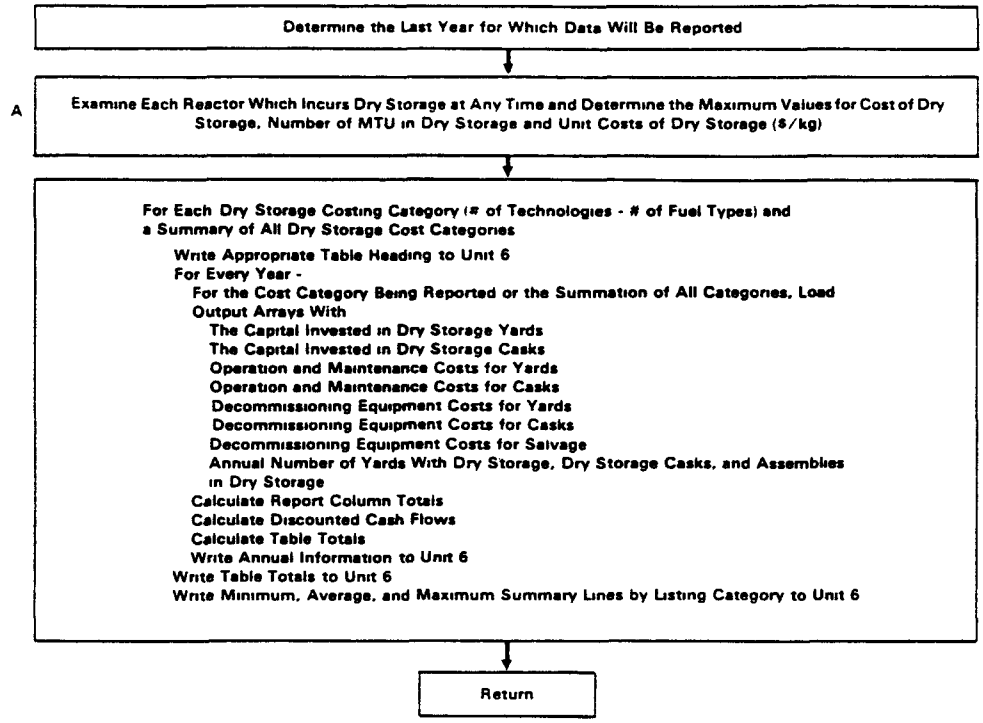


FIGURE 8.4. Block Flow Diagram of Subroutine DRYCST

TABLE 8.1. Dry Storage Cost Categories

1	Metal Cask PWR
2	Metal Cask BWR
3	Transportable Cask PWR
4	Transportable Cask BWR
5	Water Basin Storage PWR
6	Water Basin Storage BWR
7	Drywell Storage PWR
8	Drywell Storage BWR
9	Concrete Cask PWR
10	Concrete Cask BWR
11	Vault Storage PWR
12	Vault Storage BWR

writes these results and the capital, operation and maintenance, and decommissioning costs calculated by ADRCST to unit 6.

Subroutine ADRCST is called at the end of each year to calculate the dry storage costs for the year. There are four areas in which costs are determined when calculating dry storage costs:

1. the capital costs of dry storage casks and yards
2. the incremental costs for additional dry storage capacity
3. the costs to unload dry storage casks and decommission dry storage yards, and
4. the annual costs to maintain the same amount of dry storage.

Each of these four areas is discussed below in greater detail with the algorithms used to calculate the costs included in each of the four sections. Dry storage costing can be performed on a current or constant dollar basis.

If additional dry storage casks are required, the capital costs of the dry storage casks and dry storage yards are accounted for as well as the expense costs of loading the dry storage cask and property taxes. In the case of dry storage casks and yards, the capital expense of the casks and yards must be applied before the additional capacity is required.

Algorithm for Calculating Capital Costs of Dry Storage Casks and Yards

Step 1. Calculate current year expense cost

```
      TMP = (NX-LX) * LODCST(DTP)
```

Step 2. Calculate capital cost of additional dry storage

```
      TMP1 = (NX-LX) * CSTCSK(DTP) + NADYRD * ADDYRD(DTP)
```

Step 3. Sum up constant dollars

```
      DDOLRS(I,1) = DDOLRS(I,1) + TMP + TMP1
```

Step 4. Sum up current dollars

```
      DDOLRS(I,2) = DDOLRS(I,2)
     1                + TMP / CDSCNT
     2                + TMP1 * (1.0 + DISCNT) / CDSCNT
```

Step 5. Sum up taxable capital

```
      DCAPTX(I) = DCAPTX(I) + TMP1
      DCAPMT(I) = DCAPMT(I) + TMP1
```

Step 6. If this is not the first year of dry storage

```
      IF (IY .NE. 1) THEN
```

Step 7. Calculate property taxes on capital

```
       TMP = TMP1*PTAXES(DTP)
```

Step 8. Sum up constant dollars

```
       DDOLRS(I,1) = DDOLRS(I,1)
     1                 + TMP
```

Step 9. sum up current dollars

```
       DDOLRS(I,2) = DDOLRS(I,2)
     1                 + TMP * (1.0 + DISCNT) / CDSCNT
      ENDIF
```

Step 10. If this is the first year of dry storage

```
      IF( LX.EQ.0 ), THEN
```

determine how many years backward the dry storage yard capital costs can be spread for discounting

```
        II = MIN(7,IY-1)
```

Step 11. If costs can not be spread backward, skip this section

```
      IF (II .EQ. 0) GOTO 41
```

Step 12. For each year in that capital costs can be moved backwards

```
      DO 40 J = 1, II
```

determine % of capital spent in that year on dry storage yard

```
       TMP = PCT(J,DTP) * CSTYRD(DTP)
```

Step 13. Determine property taxes based on the cumulative % of capital spent up to that point

```
       TMP1 = PCTSUM(J,DTP) * CSTYRD(DTP) * PTAXES(DTP)
```

Step 14. Sum up current year capital cost and property taxes

```
       TMP = TMP + TMP1
```

Step 15. Sum up constant dollars

```
       DDOLRS(I,1)    = DDOLRS(I,1)
     *                    + TMP
```

Step 16. Sum up current dollars

```
       DDOLRS(I,2)    = DDOLRS(I,2)
     *                  + TMP * ((1.0 + DISCNT) ** J / CDSCNT)
   40          CONTINUE
   41            DCAPTX(I) = DCAPTX(I) + CSTYRD(DTP)
                 DCAPMT(I) = DCAPMT(I) + CSTYRD(DTP)
             ENDIF
```

where:

- NX — number of casks required in the current year
- LX — number of casks required in the previous year
- NADYRD — number of dry storage yards to add
- ADDYRD — contains the costs to add a dry storage yard at a reactor. It is subscripted by the dry storage cost category (see Table 8.1).
- CSTCSK — contains the capital cost to purchase a dry storage cask. It is subscripted by the 12 dry storage cost categories shown in Table 8.1.

CSTYRD contains the capital cost for the initial set up of a new dry storage yard. It is subscripted by the 12 dry storage categories shown in Table 8.1.

DCAPMT contains capital costs at each reactor. These capital costs are later multiplied by an appropriate factor to obtain maintenance costs. It is subscripted by offset WASTES reactor IDs.

DCAPTX contains capital costs at each reactor. These capital costs are later multiplied by the appropriate factor to obtain the amount of property taxes. It is subscripted by offset WASTES reactor IDs.

DDOLRS contains the cumulative total costs for dry storage at each reactor. The primary subscript is offset WASTES reactor IDs. The secondary subscript is as follows: 1 = Current Dollars, 2 = Constant Dollars.

LODCST contains the costs to load a dry storage cask. It is indexed according to the 12 dry storage costing categories shown in Table 8.1.

PCT contains the percentages of capital that must be expended in the year in which dry storage at a reactor is incurred and in the preceding six years.

PCTSUM contains the cumulative percentages of capital that must be expended in the year in which dry storage at a reactor is incurred and in the preceding six years.

PTAXES contains the property taxes for each dry storage cost category expressed as a percentage of capital. It is subscripted by the dry storage cost categories shown in Table 8.1.

Of the six types of "dry storage" technologies that can be modeled, vault storage and water basin storage have incremental capital costs as incremental capacity is added. Costs for incremental expansion of these dry storage yards are calculated as shown below.

Algorithm for Calculating Incremental Costs for Additional Dry Storage Costs

Step 1. If this is a type of dry storage that has an incremental cost, add to the initial construction money

```
     IF (VYRDCN(DTP) .NE. 0.) THEN
```

Step 2. Determine how many years backward the dry storage yard capital costs can be spread for discounting

```
       II = MINO(7,IYY-1)
```

Step 3. Calculate additional capital cost

```
       VARYRD = VYRDCN(DTP) * (NX ** VYRDEX(DTP)
     *                               - LX ** VYRDEX(DTP))
```

Step 4. If costs can not be spread backward, skip this section

```
       IF (II .EQ. 0) GOTO 51
  ZDSCNT = (1.0 + DISCNT) ** (IYY+ISTIME-1-IBYR)
```

Step 5. For each year in that capital costs can be moved backwards

```
       DO 50 J = 1, II
```

Step 6. Determine % of capital spent in that year on dry storage yard

```
        TMP = PCT(J,DTP) * VARYRD
```

Step 7. Determine property taxes based on the cumulative % of capital spent up to that point

```
        TMP1 = PCTSUM(J,DTP) * VARYRD * PTAXES(DTP)
```

Step 8. Sum up current year capital cost and property taxes

```
        TMP  = TMP + TMP1
```

Step 9. Sum up constant dollars

```
        DDOLRS(I,1)    = DDOLRS(I,1)
     *                 + TMP
```

Step 10. Sum up current dollars

```
        DDOLRS(I,2)    = DDOLRS(I,2)
     *                 + TMP * ((1.0 + DISCNT) ** J / ZDSCNT)
   50       CONTINUE
```

Step 11. Add in property taxes and maintenance costs for every year since the reactor first incurred dry storage

```
51          IF (IYY .EQ. IY) GOTO 53
            DO 52 J = IYY, IY-1
   ZDSCNT = (1.0 + DISCNT) ** (J - 1 + ISTIME - IBYR)
```

Step 12. Calculate property taxes and maintenance costs as a % of incremental capital

```
            TMP    = VARYRD * (PTAXES(DTP) + MAINT(DTP))
```

Step 13. Sum the constant dollars

```
            DDOLRS(I,1)    = DDOLRS(I,1)
  *                        + TMP
```

Step 14. Sum the current dollars

```
            DDOLRS(I,2)    = DDOLRS(I,2)
  *                        + TMP / ZDSCNT
52          CONTINUE
53            DCAPTX(I) = DCAPTX(I) + VARYRD
              DCAPMT(I) = DCAPMT(I) + VARYRD
          ENDIF
```

where: VYRDCN contains the incremental dry storage yard cost per dry storage canister in excess of 1000 casks.

VYRDCN contains a exponent used in calculating the incremental dry storage yard cost per dry storage canister in excess of 1000 casks.

In decommissioning a dry storage yard, three types of costs are addressed. The costs are: 1) the cost to unload the dry storage casks, 2) the cost to decommission the dry storage cask, and 3) the cost to decommission the dry storage yard. It is assumed that no salvage revenues will be obtained in the decommissioning process.

Algorithm for Calculating Costs to Unload Dry Storage Casks and Decommission Dry Storage Yards

Step 1. If there is no dry storage, close the yard

```
          IF( NX.EQ.0 ) THEN
```

Calculate the cost to decommission the yard for the maximum number of dry storage casks that were even in the yard

```
                TMP = CSTDCM(DTP) * CSTYRD(DTP)
     *            + CSTDCM(DTP) * VYRDCN(DTP) * NXMAX(I) ** VYRDEX(DTP)
```

Step 2. Sum the constant dollars

```
                DDOLRS(I,1) = DDOLRS(I,1)
     1                      + TMP
```

Step 3. Sum the current dollars

```
                DDOLRS(I,2) = DDOLRS(I,2)
     1                      + TMP / CDSCNT / (1.0 + DISCNT)
```

Step 4. Calculate the cost to decommission the yard for the maximum number of dry storage casks that were even in the yard

```
                TMP = NXMAX(I) * CSTDCM(DTP) * CSTCSK(DTP)
     *            + (NXMAX(I)/NCSKYD(DTP)) * CSTDCM(DTP) * VYRDEX(DTP)
```

Step 5. Sum the constant dollars

```
                DDOLRS(I,1) = DDOLRS(I,1)
     1                      + TMP
```

Step 6. Sum the current dollars

```
                DDOLRS(I,2) = DDOLRS(I,2)
     1                      + TMP / CDSCNT / (1.0 + DISCNT)
              ENDIF
```

Step 7. Calculate unload cost of casks in constant dollars

```
            DDOLRS(I,1) = DDOLRS(I,1) + (LX-NX) * UNLDCST(DTP)
```

Step 8. Calculate unload cost of casks in current dollars

```
            DDOLRS(I,2) = DDOLRS(I,2) + (LX-NX) * UNLDCST(DTP)
     1                  / CDSCNT
           ENDIF
         ENDIF
```

where: CSTDCM contains the cost to decommission the dry storage yard. It is subscripted by the 12 dry storage cost categories shown in Table 8.1.

NCSKYD contains the maximum number of dry storage casks that can be contained in each type of dry storage yard. It is subscripted according to the dry storage cost categories shown in Table 8.1

NXMAX contains the maximum number of dry storage casks in use at each reactor at any time. This is used to reuse of dry storage casks that may be at the reactor site and empty.

UNLDCST contains the costs to unload a dry storage cask. It is indexed according to the 12 dry storage costing categories shown in Table 8.1

The costs for annual maintenance of dry storage occur in three areas. These areas are labor, property taxes, and maintenance. In the case of the maintenance cost calculation for the year in which dry storage was emptied, the capital costs from the previous year are used to estimate the maintenance costs for the last year. In other years, the capital costs in the year being calculated are used to estimate the maintenance costs.

Algorithm for Calculating the Annual Costs to Maintain the Same Amount of Dry Storage

Step 1. Calculate labor cost in constant dollars

```
DDOLRS(I,1) = DDOLRS(I,1) + NX * CSTLAB(DTP)
```

Step 2. Calculate labor cost in current dollars

```
DDOLRS(I,2) = DDOLRS(I,2) + NX * CSTLAB(DTP)
1                / CDSCNT
```

Step 3. Calculate property taxes in constant dollars

```
DDOLRS(I,1) = DDOLRS(I,1) + DCAPTX(I) * PTAXES(DTP)
```

Step 4. Calculate property taxes in current dollars

```
DDOLRS(I,2) = DDOLRS(I,2) + DCAPTX(I) * PTAXES(DTP)
1                / CDSCNT
IF (NX .EQ. 0) THEN
```

Step 5. Calculate maintenance costs in constant dollars for the year in which dry storage was emptied

```
DDOLRS(I,1) = DDOLRS(I,1) + DCAPTX(I) * MAINT(DTP)
```

Step 6. Calculate maintenance costs in current dollars for the year in which dry storage was emptied

```
            DDOLRS(I,2) = DDOLRS(I,2) + DCAPTX(I) * MAINT(DTP)
     1                  / CDSCNT
          ELSE
```

Step 7. Calculate maintenance costs in constant dollars for years that have dry storage

```
            DDOLRS(I,1) = DDOLRS(I,1) + DCAPMT(I) * MAINT(DTP)
```

Step 8. Calculate maintenance costs in current dollars for years that have dry storage

```
            DDOLRS(I,2) = DDOLRS(I,2) + DCAPMT(I) * MAINT(DTP)
     1                  / CDSCNT
          ENDIF
        ENDIF
        DCAPMT(I) = DCAPTX(I)
        DCAPTX(I) = DCAPTX(I) - DECOM
```

where: CSTLAB contains the labor costs for dry storage casks. It is subscripted according to the dry storage cost categories shown in Table 8.1.

MAINT contains the maintenance costs for dry storage casks and yards as a percentage of capital. It is subscripted according to the 12 dry storage cost categories shown in Table 8.1

9.0 UTILITY ROUTINES

The utility routines used by WASTES are described below.

ACCEPT Subroutine ACCEPT determines if a particular batch of fuel has met all age acceptance requirements. WASTES supports different minimum fuel-age and residence requirements for each facility (including pools and dry storage). In the case of optimized facilities, the fuel must meet the acceptance requirements for both facilities.

ADRCST Subroutine ADRCST calculates the annual costs of dry storage.

AGECAT Integer function AGECAT returns the proper array index for one of the seven age-characterization categories to which a batch belongs.

BANNER Subroutine BANNER writes the WASTES banner to unit 6 (the .PRT file). The banner page contains the version number, release number, and other information about the executable.

BOOBOO Subroutine BOOBOO is an error processor for errors detected in the format of the WASTES input cards.

CHGALT Subroutine CHGALT changes the alternate search priorities by which facilities "simultaneously" solicit fuel.

CHGCAP Subroutine CHGCAP changes the receipt and unloading capacities of the destination/processing facilities being modeled.

CHGPRI Subroutine CHGPRI changes the search priorities by which facilities "sequentially" solicit fuel.

CHKDRY Subroutine CHKDRY is an error-checking routine that determines if dry storage is occurring in integer multiples of the appropriate dry storage cask capacities.

CHKINT Subroutine CHKINT is an error-checking routine that determines if shipments from reactors are occurring in integer multiples of the appropriate shipping cask capacity.

COSTER Subroutine COSTER calls the subroutine that writes dry storage costs to unit 6.

CPYFROM Subroutine CPYFROM decodes the compressed batch attributes from the BATCH array and copies them into the WRKBAT array.

CPYTO Subroutine CPYTO copies the batch attributes contained in the WRKBAT array into the BATCH array.

CSKUSE Integer Function CSKUSE determines whether a facility uses a cask as a shipping cask only or as a receiving cask only or for both functions.

CVIC Subroutine CVIC decodes the free form fields from the WASTES input cards. The values as well as indicators as to the type of field that was read in (character, numeric) are returned.

DISCALC Subroutine DISCALC generates a table of the distances from each reactor to each facility and from every management/processing facility to every other management/processing facility.

DISCLAIM Subroutine DISCLAIM writes a disclaimer to unit 6.

DUMPQ Subroutine DUMPQ was written to assist the programmer in debugging WASTES. It is intended to be called from the interactive debugger. The function of the subroutine is to examine a WASTES queue, convert each batch pointer into the WRKBAT array, and write the results in the file NPRNT.

ECHO Subroutine ECHO reads the WASTES input cards from NCRDR unit 5 and echos the cards with a sequential identification (ID) number to unit 6. Additionally, WASTES input cards are written to unit CREADR unit 2 for later use by subroutine CVIC.

EVENT Subroutine EVENT is called by SLAM according to events scheduled on SLAM's event calendar. Subroutine EVENT turns control over to the appropriate WASTES subroutine for processing this type of event. Events that are processed include individual reactor discharges, initiation of reactor decommissioning, initiation of facility-solicited fuel movements, changes in facility-operations schedules, changes in facility-receipt priorities, and enabling or disabling transshipments.

EXPAND Subroutine EXPAND converts the major/minor/generic facility ID scheme used on the PRIOR and ALTPRI input cards into the appropriate internal WASTES IDs.

EXPCAT Integer Function EXPCAT returns the proper array index for one of the six exposure characterization categories to which a batch belongs.

EXPRXID Subroutine EXPRXID expands the ranges of reactors to be processed (REACTORS card) and not to be processed (NOTRXIDS card) into an integer array with flags indicating whether or not the reactor is to be processed.

FACORDR Subroutine FACORDR sets up an array containing the facility IDs within classes of facilities. The facility IDs are ordered according to increasing geographic distance from the source (e.g., the first ID is the closest facility). This subroutine is called whenever the proximal shipping algorithm is being used.

FCINVC Subroutine FCINVC characterizes the fuel located at waste management/processing facilities according to age, heat generation rate, and exposure. The results are stored in arrays that are written to unit 1 at the end of the simulation. This is an annual characterization and is only called if characterization is specified by the user.

FCSHPC Subroutine FCSHPC loads the facility shipping arrays with the minimum, average, and maximum characterizations for all facility shipments. This is an annual characterization and is only called if characterization is specified by the user.

FETCSK Subroutine FETCSK determines the cask ID of the highest priority shipping cask that is acceptable to the source as a shipping cask and to the destination as a receiving cask. If no cask is located, a cask ID of zero is returned. The cask IDs are monotonically increasing IDs that are assigned in the order the cask cards are read in (1st cask = 1, 2nd cask = 2, etc.).

FNDFST Integer Function FNDFST locates the oldest batch in the pool or dry storage of the reactor that is being decommissioned.

GENTRP Function GENTRP performs a second order interpolation/extrapolation of tabular heat data to determine the current heat generation rate of a batch of fuel.

GETLOW Subroutine GETLOW gets the oldest entity from all of the possible sources that are flagged in array FACDAT. The attributes of this oldest batch are then loaded into the WRKBAT array.

HFUNC Function HFUNC determines the current heat generation rate of batches of spent fuel. HFUNC initializes tabular heat generation rate data and calls GENTRP to perform the interpolation/extrapolation of the data.

HRANGE Integer Function HRANGE returns the proper array index for one of the seven heat-generation rate-characterization categories to which a batch belongs.

IINV Function IINV returns the correct inventory and shipping array subscript given the facility ID number.

MRKTIM Subroutine MRKTIM writes the current simulated time (TNOW) to unit 6. It is intended to be called from an interactive debugger.

MV2POOL Subroutine MV2POOL moves assemblies from dry storage at a reactor back into the pool. This is typically done when some dry storage at a reactor has been unloaded leaving a non-integer amount of fuel in dry storage. The excess amount is moved back into the shipping racks of the pool.

OPTFUNC Function OPTFUNC returns value that the optimized working queue is sorted on. Depending on the type of optimization requested by the user, this value is the difference in distances between the two optimized facilities or the difference in shipping costs between the two optimized facilities.

PRXSHP Subroutine PRXSHP writes the output files that contain the annual shipment summaries. A biannual summary of reactor shipments in assemblies is written to unit FOR070. A biannual summary of reactor shipments in MTU is written to unit FOR096. An annual summary of reactor shipments in assemblies is written to FOR095.

PUTBAT Subroutine PUTBAT encodes and copies the attributes from the WRKBAT array to the next available slot in the BATCH array. This creates a new batch (sub-batch) and is used when new batches are received into the simulation or when an existing batch must be split.

REMOVE Subroutine REMOVE unlinks a batch pointer from a queue. Remove is most often used when processing optimized shipments by removing them from the source queue before they are linked to the optimized work queue.

ROUND Subroutine ROUND rounds the number of assemblies that a facility has capacity to accept to an integer cask quantity. If the facility's capacity is an integer cask amount, the quantity is unchanged. If the facility's capacity is not an integer cask quantity, the next smallest integer cask number of assemblies is returned.

RXINVC Subroutine RXINVC characterizes the fuel at reactors (both pool and dry storage) into the seven heat and age categories and the six exposure categories. RXINVC is called only if the user has requested characterization (CHARACTERIZE,YES).

RXSHPC Subroutine RXSHPC writes the minimum, average, and maximum ages, and the heat and exposure characterizations of reactor shipments to accounting arrays annually. Reactor shipments include shipments from both reactor pools and dry storage but do not include transshipments between reactors. RXSHPC is called only if the user has requested characterization (CHARACTERIZE,YES).

SETORDR Subroutine SETORDR sets the SLAM sort order for SLAM queue numbers IBEG through IEND.

SETRNG Subroutine SETRNG searches the array of reactors that are to be included in the simulation (from either end) and returns the first true value in variable IRX.

SETUP Subroutine SETUP initializes the PRIOR and ALTPRI arrays. These arrays respectively contain the default priorities for the sequential order in which a facility looks to other facilities and the facilities that are simultaneously examined for the oldest fuel.

SETVAL Subroutine SETVAL is most often used to zero-out arrays. Functionally, it sets a specified number of values in a specified array to a specified value.

SHELL Subroutine SHELL performs a shell-sort for single-dimension real arrays and returns an array of re-sorted indices.

SHELLM Subroutine SHELLM performs a shell-sort for a two-dimensional real array and returns an array of re-sorted indices.

SHELLMSM Subroutine SHELLMSM performs a shell-sort for a single-dimension real array.

SPLIT Subroutine SPLIT splits a batch into two sub-batches.

UPDECOM Subroutine UPDECOM captures annual information on reactors that are past their shutdown date as read in from the reactor-site data file.

UPDHT Subroutine UPDHT updates the heat generation rate for each entry in the batch array annually. To increase the efficiency of this operation, the batch array is accessed directly.

UPDSRC Subroutine UPDSRC updates the capacities of the source facility.

VERIFY Subroutine VERIFY is used to verify that the file name on the first line of the reactor site, discharge, and truck and rail shipping distance data files match the file name as requested by the user or as initialized in BLOKDATA.

WBAT Function WBAT accesses the batch array directly and returns the value of the I batch attribute of J batch ID. If only a single attribute is required, this function is substantially more efficient than the otherwise required pair of calls to COPY and CPYFROM.

WRTMRS Subroutine WRTMRS writes information to FORTRAN unit 98 for use by the stand-alone MRS cost model.

WRTSHP Subroutine WRTSHP writes a record of each batch that is transferred to FORTRAN unit 99. This file is used as a driver for the SIMAN disbatching model and for various post processors.

WRTWRK Subroutine WRTWRK writes the contents of the WRKBAT array to unit NPRNT (FOR006 - .PRT).

10.0 REFERENCES

Libby, R. A. and G. M. Holter. 1984. "Near-Term Decay Heat and Age of Spent Fuel in Commercial Power Reactor Inventories." In Waste Management 84. (ed. R. Post). Proceedings of the Symposium on Waste Management, Tucson, Arizona.

Merrill E. T. and J. F. Fletcher. 1983. Economics of At-Reactor Spent Fuel Storage Alternatives. PNL-4517, Pacific Northwest Laboratory, Richland, Washington.

Shay, M. R. and M. E. Buxbaum. 1986. WASTES: Waste System Transportation and Economic Simulation - Version II, Users Guide. PNL-5714, Pacific Northwest Laboratory, Richland, Washington.

Pritsker, A. A. B. and C. D. Pegden. 1979. Introduction to Simulation and SLAM. Systems Publishing Corp., West Lafayette, Indiana.

U.S. Department of Energy (DOE). 1983. Spent Fuel Storage Requirements. DOE/RL-83-1, Richland Operations Office, Richland, Washington.

APPENDIX A

WASTES SUBROUTINE/FUNCTION DESCRIPTIONS

APPENDIX A

WASTES SUBROUTINE/FUNCTION DESCRIPTIONS

ACCEPT Subroutine ACCEPT determines if a particular batch of fuel has met both age acceptance requirements. The requirements are that the fuel has resided at the previous location long enough and is old enough to be acceptable at the destination facility. In the case of optimized facilities, the fuel must be old enough to be acceptable to both facilities.

ACCEPT (IFR, L, AXEPT1, AXEPT2, OPT, BXEPT1)

IFR	(I*4)	ID of current physical location of batch.
L	(I*4)	ID of destination.
AXEPT1	(L*4)	If TRUE, indicates that fuel is old enough for acceptance at destination facility.
AXEPT2	(L*4)	If TRUE, indicates that fuel has remained at the current location the required number of years.
OPT	(L*4)	If TRUE, indicates that the destination is an optimized class of facility.
BXEPT1	(L*4)	Not currently implemented.

ADRCST Subroutine ADRCST calculates the annual costs of dry storage.

ADRCST

AGECAT Integer Function AGECAT returns the proper array index for one of the seven age characterization categories to which a batch belongs.

INTEGER FUNCTION AGECAT(AGE)

AGE	(R*4)	Age of fuel in decimal years.

BANNER Subroutine BANNER writes the WASTES banner to the FOR006 (.PRT) file. The banner page contains the version #, release #, and other information about the executable.

BANNER (NPRNT, LINE1, LINE2, LINE3)

NPRNT	(I*4)	FORTRAN unit number for which the banner is to be written.

LINE1 (CHAR*1) Contains up to 11 characters that are to be written to the first line of unit NPRNT.

LINE2 (CHAR*1) Contains up to 11 characters that are to be written to the second line of unit NPRNT.

LINE3 (CHAR*1) Contains up to 11 characters that are to be written to the third line of unit NPRNT.

BIOFMT Subroutine BIOFMT empties the optimized holding queues of all classes of optimized facilities at the end of the year.

BIOFMT (FACTYP)

FACTYP (I*4) The ID of the optimized class of facility whose holding queue is to be emptied.

BOOBOO Subroutine BOO BOO is an error processor for errors detected in the format of the WASTES input cards.

BOOBOO (NUM , KFLD , IALPH , CRDNUM, VALUE, NTRM)

NUM (I*4) Error number that determines which error format to use.

KFLD (I*4) Number of the input field on the card that is in error.

IALPH (C*20) KEYWORD input card name.

CRDNUM (I*4) Sequential number of input card (as read in).

VALUE (R*4) Array of input values as deciphered by CVIC.

NTRM (I*4) Array of mode values as set by CVIC.

CHGALT Subroutine CHGALT changes the alternate search priorities by which facilities "simultaneously" solicit fuel.

CHGALT

CHGCAP Subroutine CHGCAP changes the receipt and unloading capacities of the destination/processing facilities being modeled.

CHGCAP

CHGPRI Subroutine CHGPRI changes the search priorities by which facilities "sequentially" solicit fuel.

CHKDRY Subroutine CHKDRY is an error checking routine that determines if dry storage is occurring in integer multiples of the appropriate dry storage cask capacities.

CHKDRY

CHKINT Subroutine CHKINT is an error checking routine that determines if shipments from reactors are occurring in integer multiples of the appropriate shipping cask capacities.

CHKINT

COSTER Subroutine COSTER calls the subroutine that writes dry storage costs to the file NPRNT (FOR006 - .PRT).

COSTER (MAJTBL)

MAJTBL (I*4) Major ID of the nest table to be written to file NPRNT (FOR006 - .PRT).

CPYFROM Subroutine CPYFROM copies the compressed batch attributes from the BATCH array into the WRKBAT array.

CPYFROM (IPTR)

IPTR (I*4) Pointer to BATCH array.

CPYTO Subroutine CPYTO copies the batch attributes contained in the WRKBAT array into the BATCH array.

CPYTO (NTRY)

NTRY (I*4) Pointer to BATCH array.

CSKUSE Integer Function CSKUSE determines whether a cask is used as a shipping cask only or as a receiving cask only or if it can be used for both functions.

INTEGER FUNCTION CSKUSE(NAMCHR)

NAMCHR (C*1) Name of cask for which determination is desired.

CVIC Subroutine CVIC decodes the free form fields from the WASTES input cards. The values and the indicators of the type of field, which is read in (character, numeric), are returned.

CVIC (KARD , NFLD , NTAPE , VALUE , NTRM, IERROR)

KARD (I*4) Contains up to a maximum of 53 free form fields to be interpreted by CVIC.

NFLD (I*4) Number of fields which were decoded by CVIC.

NTAPE (I*4) FORTRAN unit to which alpha fields are written by CVIC.

VALUE (R*4) Array containing numeric values decoded by CVIC.

NTRM (I*4) Field type indicator array (+ values indicate numeric field, - values indicate character field).

IERROR (I*4) Not currently implemented.

DECOM Subroutine DECOM attempts to decommission (empty the pool and dry storage) reactors. This is done for each reactor when the current time is greater than the decommissioning date plus the user supplied decommissioning lag time. Since subroutine DECOM executes at year.95, it has priority over facility solicited material movements and less priority than forced discharge events.

DECOM

DISCALC Subroutine DISCALC generates a table of the distances from each reactor to each facility and from every management/processing facility to every other management/processing facility.

DISCALC

DISCLAIM Subroutine DISCLAIM writes a disclaimer to NPRNT (FOR006 - .PRT file)

DISCLAIM (NPRNT)

NPRNT (I*4) The FORTRAN unit disclaimer.

DRYCST Subroutine DRYCST writes the dry storage cost reports to NPRNT (FOR006 - .PRT).

DRYCST (MAJTBL, MINTBL)

MAJTBL (I*4) The major table ID of the next table to be written.

MINTBL (I*4) The minor table ID of the next table to be written.

DSBCH Subroutine DSBCH moves batches of material from reactors having an FCR discharge to the destination facility. If the shipment did not result in an integer shipping cask amount of material, a logical flag is set requiring RXDIS to obtain more material to fill the cask.

DSBCH (IMOVE, ISTOR, IDTO, MARKED, NALFT, OPT, CSKASMS)

IMOVE	(I*4)	The number of assemblies in the current batch when DSBCH is called.
ISTOR	(I*4)	The pool in which the batch is located when DSBCH is called (if it is a shared pool, ID is the lower ID of the shared pair).
IDTO	(I*4)	WASTES ID of the destination facility. In the case of optimized facilities, IDTO is the class of facility to be optimized.
MARKED	(L*4)	If TRUE, it indicates that the batch transferred by DSBCH did not make an integer cask shipment and more material is required.
NALFT	(I*4)	The number of assemblies left to move.
OPT	(L*4)	If TRUE, shipment is to an optimized facility class.
CSKASMS	(I*4)	The number of assemblies in the shipping cask used between the reactor and the destination facility.

DUMPQ Subroutine DUMPQ was written to assist programmers in debugging WASTES. It is intended to be called from the interactive debugger. The function of the subroutine is to examine a WASTES queue, convert each batch pointer into the WRKBAT array, and write the result the file NPRNT.

DUMPQ (IDQ)

IDQ	(I*4)	WASTES ID of queue to be dumped.

ECHO Subroutine ECHO reads the WASTES input cards from unit NCRDR (FOR005) and echos the cards with a sequential id number to unit NPRNT (FOR006). Additionally, WASTES input cards are written to unit CREADR (FOR002) for later use by subroutine CVIC.

ECHO (NCRDR , CREADR, NPRNT)

NCRDR	(I*4)	Input card unit (FOR005 - .INP)
CREADR	(I*4)	Work unit for use by CVIC (FOR002)
NPRNT	(I*4)	"Print" output unit (FOR006 - .PRT)

EVENT Subroutine EVENT is called by SLAM according to events scheduled on SLAM's event calendar. Subroutine Event turns control over to the appropriate WASTES subroutine for processing this type of event. Events that are processed include reactor discharges, reactor decommissioning, facility-solicited fuel movements, changes in facility operations schedules, changes in facility receipt priorities, and enabling or disabling transshipments.

EVENT (IX)

IX	(I*4)	Event code for event to be processed.

EXPAND Subroutine EXPAND converts the major/minor/generic facility ID scheme used on the PRIOR and ALTPRI input cards into the appropriate internal WASTES IDs.

EXPAND (FACARA ,NUMINPS ,IFARAY ,NRETND)

FACARA	(R*4)	Contains the floating point facility IDs as specified by the user in the PRIOR and ALTPRI cards.
NUMINPS	(I*4)	Contains the number of facility IDs contained in array FACARA.
IFARAY	(I*4)	Contains the internal WASTES facility IDs.
NRETND	(I*4)	Contains the number of facility IDs that are translated.

EXPCAT Integer Function EXPCAT returns the proper array index for one of the six exposure characterization categories to which a batch belongs.

INTEGER FUNCTION EXPCAT (EXP)

EXP	(R*4)	Exposure of fuel in MWD/MT.

EXPRXID Subroutine EXPRXID expands the ranges of reactors to be processed (REACTORS card) and not to be processed (NOTRXIDS card) into an integer array with flags indicating whether or not the reactor is to be processed.

EXPRXID (RXIDS , NRXIDS, RUNIDS, NPRNT , ERROR)

RXIDS	(I*4)	Array containing unexpanded ranges of reactors that will or will not be processed (-1 indicates reactor will be processed).
NRXIDS	(I*4)	Number of fields to be decoded.

RUNIDS (I*4) Array containing flag indicating whether or not a reactor is to be processed.

NPRNT (I*4) "Print" output unit (FOR006 - .PRT).

ERROR (L*4) If TRUE, an error was detected in expanding the reactor IDs, and WASTES execution is terminated.

FAC Subroutine FAC attempts to fill the unused annual receipt capacity of all waste management facilities. It does this by calling the appropriate subroutine to process eachass of facilities according to the correct shipping algorithm. Since FAC runs at the end of the year, a number of housekeeping operations are also carried out. The annual receipt capacities of all facilities are updated, annual shipping and inventory and shipping arrays are written and reinitialized, the error checking routines are called, etc.

FAC (USRMTR, MSTRT, INCMON)

USRMTR (L*4) Not Currently Implemented.

MSTRT (I*4) Not Currently Implemented.

INCMON (I*4) Not Currently Implemented.

FACORDR Subroutine FACORDR sets up an array containing the facility IDs within classes of facilities. The facility IDs are ordered according to increasing geographic distance from the source (e.q., the first ID is the closest facility). This subroutine is called whenever the proximal shipping algorithm is being used.

FACORDR (ITYPE , ISTOR , INORDR)

ITYPE (I*4) Facility type that is to be ordered.

ISTOR (I*4) The offset WASTES ID for the current physical location of the batch for which the determination is required.

INORDR (I*4) Array containing the ordering of facilities in this class (closest first).

FCINVC Subroutine FCINVC characterizes the fuel located at waste management/processing facilities according to age, heat generation rate, and exposure. The results are stored in arrays that are written to FOR001 at the end of the simulation. This is an annual characterization due to the calling sequence. This subroutine is only called if additional characterization is requested by the user.

FCINVC

FCSHPC Subroutine FCSHPC loads the facility shipping arrays with the minimum, the average, and the maximum characterizations for all facility shipments. This is an annual characterization due to the calling sequence. This subroutine is only called if additional characterization is requested by the user.

FCSHPC

FETCSK Subroutine FETCSK determines the cask ID of the highest priority shipping cask that is acceptable to the source as a shipping cask and that is also acceptable to the destination as a receiving cask. If no cask is located, a cask ID of 0 is returned. The cask IDs are monotonically increasing IDs that are assigned in the order in which the cask cards were read in (1st cask = 1, 2nd cask = 2, etc).

FETCSK (FROM , TO , CASK , OPT)

FROM	(I*4)	WASTES ID of source of shipment.
TO	(I*4)	WASTES ID of shipment destination.
CASK	(I*4)	ID of cask that should be used in shipment.
OPT	(L*4)	If TRUE, it indicates that the destination is an optimized facility class. The shipping cask must be acceptable to both of the optimized facilities.

FILLCSK Subroutine FILLCSK determines if the last batch that was transferred resulted in an integer cask shipment. If not, the next oldest batch is loaded into the WRKBAT array, split if necessary, and a logical flag that indicates that the batch in the WRKBAT array must be shipped is set.

FILLCSK (IFILE, N2RECV, NASMS, CSKASMS, IP2, BYHEAT, PRIDAT, ISTOR, IRE, NTRY, PARTIAL, FLAG)

IFILE	(I*4)	WASTES offset ID for source file from which material is to be moved.
N2RECV	(I*4)	The number of assemblies to be received at this destination.
NASMS	(I*4)	Number of assemblies in the next oldest batch located by FILLCSK.
CSKASMS	(I*4)	The number of assemblies in the shipping cask used between the source and this destination.

IP2 (I*4) The WASTES offset ID of the pool from which the batch was originally discharged. If the pool is shared, the ID is the lower ID of the shared pair.

BYHEAT (L*4) Not currently implemented.

ISTOR (I*4) The WASTES offset ID of the current batch location.

IRE (I*4) The WASTES offset ID of the reactor that originally discharged the batch.

NTRY (I*4) Pointer to the BATCH array for the batch that has just been transferred.

PARTIAL (L*4) If TRUE, it indicates a partial cask shipment.

FLAG (L*4) Indicates that a partial cask shipment has been made and the batch in the WRKBAT array must be transferred.

FNDFST Integer Function FNDFST locates the oldest batch in the pool or dry storage of the reactor that is being decommissioned.

INTEGER FUNCTION FNDFST (ID,INQ)

ID (I*4) The pool ID (WASTES offset) of the reactor that is being decommissioned. If the pool is shared, the ID is the lower ID of the shared pair.

INQ (I*4) Source file for material to be moved. Can be the pool ID (WASTES offset) of the reactor that is being decommissioned or the generic dry storage queue.

FRCAPO Subroutine FRCAPO attempts to fill the remaining annual receipt capacity of an optimized class of facility according to the optimized shipping algorithm. This accomplishes the optimized-facility-solicited material movements.

FRCAPO (FACTYP)

FACTYP (I*4) ID of the optimized facility class that is to be filled.

FRCAPP Subroutine FRCAPP attempts to fill the remaining annual receipt capacity of a class of facility according to the proximal shipping algorithm. This accomplishes the proximally-routed-facility solicited material movements.

FRCAPP (FACTYP)

FACTYP	(I*4)	ID of facility class that is to be filled proximally.

FRCAPS Subroutine FRCAPS attempts to fill the remaining annual receipt capacity of a class of facility according to the sequential shipping algorithm. This accomplishes the sequentially-routed-facility solicited material movements.

FRCAPS (IBASE , JBASE)

IBASE	(I*4)	ID of lowest facility in the sequentially-filled class of facilities.
JBASE	(I*4)	ID of highest facility in the sequentially-filled class of facilities.

GENTRP Function GENTRP performs a second-order interpolation/extrapolation of tabular heat data to determine the current heat generation rate of a batch of fuel.

FUNCTION GENTRP(X,Y,Z,FIJK,IM,JM,XMIN,YMIN,ZMIN,XMAX,YMAX,ZMAX,DX,DY,DZ,IFLAG)

X	(R*4)	Age of fuel in decimal years.
Y	(R*4)	Exposure of fuel in MWD/MT.
Z	(R*4)	= 0.0 Only a 2 dimensional interpolation/extrapolation is currently required as data on the heat generation rate as a function of initial enrichment has not yet been developed.
FIJK	(R*4)	Table of real values to be interpolated. Must be dimensioned IM x JM x 1.
IM	(I*4)	Primary dimension of array FIJK.
JM	(I*4)	Secondary dimension of array FIJK.
XMIN	(R*4)	Minimum value of X used to construct FIJK.
YMIN	(R*4)	Minimum value of Y used to construct FIJK.
ZMIN	(R*4)	Minimum value of Z used to construct FIJK.
XMAX	(R*4)	Maximum value of X used to construct FIJK.
YMAX	(R*4)	Maximum value of Y used to construct FIJK.

ZMAX	(R*4)	Maximum value of Z used to construct FIJK.
DX	(R*4)	X increment used in construction of FIJK.
DY	(R*4)	Y increment used in construction of FIJK.
DZ	(R*4)	Z increment used in construction of FIJK.
IFLAG	(I*4)	Boundary indicator. 0 = interior to table, 1 = exterior to table (extrapolated).

GETLOW Subroutine GETLOW gets the oldest entity from all of the possible sources that are flagged in array FACDAT. The attributes of this oldest batch are then loaded into the WRKBAT array.

GETLOW (FACDAT ,FLAG ,BYHEAT)

FACDAT	(I*4)	Array containing flags that indicate which queues should be examined when locating the oldest batch.
FLAG	(I*4)	If FLAG = 0 upon return from GETLOW, no entities were located in any of the queues that were searched (controlled by FACDAT). If GETLOW is called with FLAG negative, the next oldest batch from the same location is loaded into the batch array.
BYHEAT	(L*4)	Not Currently Implemented.

GREATC Function GREATC determines the great circle distance in miles between two latitude/longitude pairs.

FUNCTION GREATC (A ,B ,C ,D)

A	(R*4)	Latitude of point #1.
B	(R*4)	Longitude of point #1.
C	(R*4)	Latitude of point #2.
D	(R*4)	Longitude of point #2.

HFUNC Function HFUNC determines the current heat generation rate of batches of spent fuel. HFUNC initializes tabular heat generation rate data and calls GENTRP to perform the interpolation/extrapolation of the data.

FUNCTION HFUNC (N , E , A)

N	(I*4)	Type of Fuel: 1 = PWR, 2 = BWR.
E	(R*4)	Exposure of Fuel in MWD/MT.
A	(R*4)	Age of fuel (time since discharge in decimal years.)

HRANGE Integer Function HRANGE returns the proper array index for one of the seven heat generation rate characterization categories to which a batch belongs.

INTEGER FUNCTION HRANGE(HVAL)

HVAL	(R*4)	Heat generation rate of fuel in MW/MT.

IINV Function IINV returns the correct inventory and shipping array subscript given the facility ID number.

FUNCTION IINV (IA)

IA	(I*4)	WASTES facility ID number.

INTLC Subroutine INTLC performs all of the initializations necessary to begin the simulation. The largest two of these activities are structuring the simulation according to the input cards and reading and setting up the reactor site data.

INTLC

LEVEL Subroutine LEVEL calculates the shipping cask purchasing requirements if shipping casks had been purchased and used in accordance with a levelized cask purchasing philosophy. If the required number of casks in a given year is less than the number of casks available, the excess casks become available to move fuel in the current year, thus reducing the peak requirements in future years. Up to 10 percent of the total annual receipt capacity of all facilities may be advanced in this manner.

LEVEL (NCASK, AVMTPC, MTS, CLIFE , FRSTYR, LSTYR)

NCASK	(I*4)	ID of cask that is being levelized.
AVMTPC	(R*4)	Average number of MTU moved by each cask of the type being levelized.
MTS	(R*4)	Lagpool size (initialized to 300 in BLOKDAT).
CLIFE	(R*4)	Lifetime of the cask type that is being levelized.

RSTYR (I*4) The first year having any cask requirement.

LSTYR (I*4) The last year having any cask requirement.

MOV2DRY Subroutine MOV2DRY moves fuel from the pool of a reactor to the dry storage at a reactor in integer dry storage cask amounts.

MOV2DRY (IP , IDMOVE)

IP (I*4) Offset WASTES ID of pool from which material is to be removed.

IDMOVE (I*4) Number of assemblies which must be moved to restore pool FCR. May or may not be an integer dry storage cask amount.

MRKTIM Subroutine MRKTIM writes the current time (TNOW) to unit NPRNT. It is intended to be called from an interactive debugger.

MRKTIM

MV2POOL Subroutine MV2POOL moves assemblies from dry storage at a reactor back into the pool. This is typically done when some dry storage at a reactor has been unloaded leaving a non-integer amount of fuel in dry storage. The excess amount is then moved back into the shipping racks of the pool.

MV2POOL (NA, IP, IDC)

NA (I*4) The number of assemblies that must be moved from dry storage back into the pool.

IP (I*4) The offset WASTES ID of the pool to which the shipment is to take place.

IDC (I*4) Not currently implemented.

OPTFUNC Function OPTFUNC returns a value upon which the optimized working queue is sorted. Depending on the type of optimization requested by the user, this value is the difference in distances between the two optimized facilities or the difference in shipping costs between the two optimized facilities.

FUNCTION OPTFUNC(ISTOR, ID1, ID2, BATAMT, NASMS, WTYP, CTYP)

ISTOR (I*4) Offset WASTES ID of the current physical location of the batch.

ID1 (I*4) WASTES ID of the first facility of the optimized destination pair.

ID2	(I*4)	WASTES ID of the second facility of the optimized destination pair.
BATAMT	(R*4)	Amount of MTU in the batch being transferred.
NASMS	(I*4)	Number of assemblies in the batch being transferred.
WTYP	(I*4)	Waste type if the batch to be shipped (1 = PWR, 2 = BWR)
CTYP	(I*4)	ID of the shipping cask to be used for the transfer.

OTPUT Subroutine OTPUT writes the outputs from the simulation to unit NPRNT (FOR006 - .PRT) and the post processor (FOR001). Additionally, the reactor shipment summaries are written to FOR070, FOR095, and FOR096.

OTPUT

PROEVNT Subroutine PROEVNT sets up and schedules all of the events that were dictated by WASTES input cards. This includes changes in facility receipt capacities (CAPACITY card) and priorities (PRIOR and ALTPRI cards) etc.

PROEVNT (HAPENS, EVNTNAME, NROW , NEVNTS)

HAPENS	(R*4)	Array containing the information and characteristics of the event to be scheduled.
EVNTNAME	(C*20)	Input card keyword for the event being scheduled.
NROW	(I*4)	Leading dimension of HAPENS array in PROEVNT (= 10).
NEVNTS	(I*4)	Number of events of this type to be processed (secondary dimension of array HAPENS).

PRXSHP Subroutine PRXSHP writes the output files that contain the annual shipment summaries. A biannual summary of reactor shipments in assemblies is written to unit FOR070. A biannual summary of reactor shipments in MTU is written to unit FOR096. An annual summary of reactor shipments in assemblies is written to FOR095.

PRXSHP

PUTBAT Subroutine PUTBAT encodes and copies the attributes from the WRKBAT array to the next available slot in the BATCH array. This creates a new batch (sub-batch) and is used when new batches are received into the simulation or when an existing batch must be split.

PUTBAT

RDPLNT Subroutine RDPLNT reads in the reactor site data file and defines the characteristics of the reactors for use by WASTES. As part of this function, pool capacities and FCR reserve margins for shared reactors are calculated, transshipment networks are set up, pool shutdown events are scheduled, a variety of other operations associated with the startup and shutdown of reactor pools are defined.

RDPLNT (TDECOM, RXCSKS)

TDECOM	(R*4)	Decommissioning lag time supplied by the user on DECOM card (default = 1.0).
RXCSKS	(C*8)	Name of casks assigned to this reactor via the RXCASKS card. If RXCSKS is blank, RDPLNT assigns an R1, T1 cask to each reactor by default.

REMOVE Subroutine REMOVE unlinks a batch pointer from a queue. REMOVE is most often used when processing optimized shipments by removing them from the source queue before they are linked to the optimized work queue.

REMOVE (ISTOR , NTO , MTO , TIMFRM, HTFRM , NNFROM, IORIG, NNTO)

ISTOR	(I*4)	WASTES offset ID of current physical location of material.
NTO	(I*4)	Not currently implemented.
MTO	(I*4)	WASTES ID of destination.
TIMFRM	(I*4)	ID of time sorted queue representing ISTOR.
HTFRM	(I*4)	Not currently implemented.
NNFROM	(I*4)	Inventory and shipping array index for WASTES offset ID of current physical location.
IORIG	(I*4)	WASTES offset ID of reactor from which batch was originally discharged.
NNTO	(I*4)	Inventory and shipping array index for destination.

ROUND Subroutine ROUND rounds the number of assemblies for which a facility has capacity to an integer cask quantity. If the facility's capacity is an integer cask amount, the quantity is unchanged. If the facility's capacity is not an integer cask quantity, the next smallest integer cask number of assemblies is returned.

ROUND (NA, CSKASMS, NCASKS, ID, NDRY, PCPCASK)

NA	(I*4)	The total number of assemblies the destination facility can accept, or if the source is another waste management facility, the number of assemblies the other facility can unload.
CSKASMS	(I*4)	If a shipment is from dry storage, the number of assemblies in a dry storage cask. Otherwise, the number of assemblies in a shipping cask.
NCASKS	(I*4)	Not currently implemented.
ID	(I*4)	*Offset WASTES ID of current physical location* of batch.
NDRY	(I*4)	WASTES ID for the generic dry storage queue.
PCPCASK	(I*4)	Number of assemblies in the shipping cask used between the source and the destination for this fuel type.

RXDIS Subroutine RXDIS processes reactor discharge events. This entails filing the discharge into the pool of the reactor and determining if the FCR storage margin of the pool has been violated. If the FCR has been violated, RXDIS attempts to 1) move the material to waste management facilities, 2) transship the material (if allowed), or finally, 3) move the material to dry storage if the preceding options could not be used.

RXDIS (IX)

IX	(I*4)	Not currently Implemented.

RXINVC Subroutine RXINVC characterizes the fuel at reactors (both pool and dry storage) into the seven heat and age categories and the six exposure categories. RXINVC is called only if the user has requested additional characterization information (CHARACTERIZE,YES).

RXINVC

RXSHPC Subroutine RXSHPC writes the minimum, the average, and the maximum age; heat; and exposure characterizations of reactor shipments to accounting arrays annually. Reactor shipments include shipments from both reactor pools and dry storage but do not include transshipments between reactors. RXSHPC is called only if the user has requested additional characterization information (CHARACTERIZE,YES).

RXSHPC

SCHDIS Subroutine SCHDIS reads information on the next discharge from the discharge data file, stores the information concerning this discharge in the BATCH array, and schedules the time that this batch is to be accepted into the simulation. Additionally, SCHDIS makes the determination of when to stop accepting fuel if a FUELSTOP card was supplied by the user.

SCHDIS

SETORDR Subroutine SETORDR sets the SLAM sort order for SLAM queue numbers IBEG through IEND.

SETORDR (IBEG ,IEND ,IORDR ,IATRIB)

IBEG	(I*4)	ID of first queue for which the sort order is to be set.
IEND	(I*4)	ID of the last queue for which the sort order is to be set.
IORDR	(I*4)	Sets the code into SLAM array IINN, which determines SLAM file ranking (1=LVF, 2=HVF, 3=FIFO, 4=LIFO).
IATRIB	(I*4)	Sets the index of the attribute for ranking SLAM file entries if SLAM array IINV(I) = 1 or 2. In the WASTES implementation, IATRIB = 1.

SETRNG Subroutine SETRNG searches the array of reactors that are to be included in the simulation (from either end) and returns the first true value in variable IRX.

SETRNG (RUNIDS, N, FRONT, IRX)

RUNIDS	(I*4)	Array of reactor IDs that are to be processed.
N	(I*4)	Dimension of array RUNIDS.
FRONT	(L*4)	If TRUE, the array RUNIDS is searched from the front, otherwise the array is searched from the back.
IRX	(I*4)	The index of the first true value encountered in array RUNIDS. IF IRX = 0, no true values were encountered.

SETUP Subroutine SETUP initializes the PRIOR and ALTPRI arrays. These arrays respectively contain the default priorities for the sequential order in which a facility looks to other facilities and the facilities that are simultaneously examined for the oldest fuel.

SETUP

SETVAL Subroutine SETVAL is most often used to zero out arrays. Functionally, it sets a specified number of values in a specified array to a specified value.

SETVAL (ARRAY , N , VAL)

ARRAY	(R*4)	Array name for which values are to be set.
N	(I*4)	Number of values that are to be set (usually array dimension).
VAL	(R*4)	Value to which an array is to be set (usually 0).

SHELL Subroutine SHELL performs a shell-sort for single-dimension real arrays and returns an array of re-sorted indices.

SHELL (X , N , INDEX)

X	(R*4)	The vector of real numbers to be sorted.
N	(I*4)	The number of entries in vector X.
INDEX	(I*4)	The array of X indices re-sorted to reflect the changes to X.

SHELLM Subroutine SHELLM performs a shell-sort for a two-dimension real array and returns an array of re-sorted indices.

SHELLM (X , NROW , N , NS , INDEX)

X	(R*4)	The matrix of real numbers to be sorted (HAPENS).
NROW	(I*4)	The row dimension of matrix X (currently = 10).
N	(I*4)	The column dimension of matrix X.
NS	(I*4)	Step size for shell-sort (= 1).
INDEX	(I*4)	The array of indices of X re-sorted to reflect the changes to X.

SHELLMSM Subroutine SHELLMSM performs a shell-sort for a single-dimension real array.

SHELLMSM (X,N)

X	(R*4)	Real array to be sorted.
N	(I*4)	Number of entries in the array.

SHPCST Subroutine SHPCST calculates the costs for shipping casks based on the cost parameters for each type of cask and the distance traveled. SHPCST also calculates and returns the number of cask busy days required for the shipment.

SHPCST (TYPE , RAIL , DIST , PAYLOD, SHIP$, RENT, SANDS$, BDAYS)

TYPE	(I*4)	Cask ID.
RAIL	(I*4)	Rail shipment flag. (1 = truck, 2 = rail).
DIST	(R*4)	One way distance from point of origin (miles).
PAYLOD	(R*4)	Weight of fuel being transferred (MTU).
SHIP$	(R*4)	Cost of shipping fuel in dollars (actual truck or train cost based on distance and weight).
RENT	(R*4)	Cost to rent shipping cask if not owned ($).
SANDS$	(R*4)	Cost for Security and Surcharges ($).
BDAYS	(R*4)	Number of cask days required for the shipment.

SPFOUT Subroutine SPFOUT writes table series 3 thru 10 to FORTRAN units 6 (.PRT file) and 1 (post processor).

SPFOUT (MAJTBL, NRECS)

MAJTBL	(I*4)	ID of next table series to be written.
NRECS	(I*4)	Number of records to be written to FOR001 for table summaries for series 3 and 4.

SPLIT Subroutine SPLIT splits a batch making two sub-batches.

SPLIT (BMOVE , NASMS , NTRY , ISTOR)

BMOVE	(I*4)	The number of assemblies (and batch attributes) that are to be in the WRKBAT array upon return from SPLIT.
NASMS	(I*4)	The number of assemblies in the batch that is to be split.

NTRY (I*4) On call = the batch ID (IWRKBAT(6)) for the batch that is to be split.
On return = the batch ID (IWRKBAT(6)) of the batch that was split off.

ISTOR (I*4) ID of the queue in which the remainder of the batch is to be stored.

TRNLST Subroutine TRNLST attempts to obtain enough material to make a multi-cask rail shipment. If enough material can be located, TRNLST then makes the shipment to the destination facility. The number and size of casks in the shipment are user specified.

TRNLST (FROM, TO, CASK)

FROM (I*4) Offset WASTES ID for current physical location of batch.

TO (I*4) WASTES ID of destination facility.

CASK (I*4) ID of cask to be used in building the multi-cask rail shipment.

TRNSHP Subroutine TRNSHP effects transfers from reactors with FCR margin violations to other reactors in the same transshipment network. If the other reactors in the network cannot accept all of the material that must be moved, TRNSHP places the excess in dry storage. All shipments occur in integer cask amounts.

TRNSHP (NMVD)

NMVD (I*4) The number of assemblies that must be moved to restore the pool's FCR margin.

UPDECOM Subroutine UPDECOM captures annual information on reactors that are past their shutdown date as read from the reactor site data file.

UPDECOM

UPDHT Subroutine UPDHT updates the heat generation rate for each entry in the batch array annually. To increase the efficiency of this operation, the batch array is accessed directly.

UPDHT

UPDISA Subroutine UPDISA updates the inventory and shipping arrays for each batch as it is moved.

UPDISA (NTO, MTO, ILOC, NASMS, BATAMT, IRANGE, NNTO, NNFROM, LSTCSK, ISTOR, CTYPE)

NTO (I*4) The WASTES ID of the destination facility.

MTO	(I*4)	The WASTES ID of the destination facility.
ILOC	(I*4)	Offset WASTES ID of current physical location of batch.
NASMS	(I*4)	Number of assemblies in the batch.
BATAMT	(R*4)	Weight of the batch in MTU.
IRANGE	(I*4)	Array index for the heat generation rate characterization category to which the batch belongs.
NNTO	(I*4)	Inventory and shipping array index for the destination.
NNFROM	(I*4)	Inventory and shipping array index for WASTES offset ID of current physical location.
LSTCSK	(L*4)	If TRUE, indicates that the costs for this shipment should be calculated for a full cask even though the cask is not full.
ISTOR	(I*4)	The WASTES offset ID of the current physical location of the batch. (if located in a pool of a shared pair, the ID is the lower ID of the shared pair).
CTYPE	(I*4)	ID of cask used for the shipment.

UPDSRC Subroutine UPDSRC updates the capacities of the source facility.

UPDSRC (ID, IP, NASMS, LSTCSK, PARTIAL, BATAMT, IRE, IP2, L)

ID	(I*4)	Offset WASTES ID of the current physical location of the batch.
IP	(I*4)	The WASTES offset ID of the current physical location of the batch. (if located in a pool of a shared pair, the ID is the lower ID of the shared pair).
NASMS	(I*4)	Number of assemblies in the batch to be transferred.
LSTCSK	(L*4)	If TRUE, indicates that the costs for this shipment should be calculated for a full cask even though the cask is not full.

PARTIAL (L*4) If TRUE, indicates a partial cask shipment at this time.

BATAMT (R*4) Weight of the batch to be transferred in MTU.

IRE (I*4) The WASTES offset pool ID of the reactor from which the batch was originally discharged.

IP2 (I*4) The WASTES offset pool ID of the reactor from which the batch was originally discharged (if located in a shared pool, the ID is the lower ID of the shared pair).

L (I*4) Not currently implemented.

VERIFY Subroutine VERIFY is used to verify that the file name on the first line of the reactor site, discharge, and truck and rail shipping distance data files match the file name as requested by the user or as initialized in BLOKDATA.

VERIFY (NAME , STRING, LEN , OKAY)

NAME (C*20) File name requested by the user or initialized in BLOKDATA.

STRING (C*70) First line of above mentioned files.

LEN (I*4) Number of characters of STRING that must match NAME.

OKAY (L*4) If TRUE, NAME and first LEN characters of STRING match.

WBAT Function WBAT accesses the batch array directly and returns the value of the Ith batch attribute of J batch ID. If only a single attribute is required, this function is substantially more efficient than the otherwise required pair of calls to COPY and CPYFROM.

FUNCTION WBAT(I, J)

I (I*4) Number of batch attribute whose value is to be returned.

J (I*4) Batch ID of the batch whose value is to be returned.

WRTMRS Subroutine WRTMRS writes out information to FORTRAN unit 98 for use by the stand-alone MRS cost model.

WRTMRS

WRTSHP Subroutine WRTSHP writes a record to FORTRAN unit 99 of each batch that is transferred. This file is used as a driver for the SIMAN dispatching model and for various post processors.

WRTSHP (ILOC, NTO, CTYPE)

ILOC	(I*4)	Offset WASTES ID of current physical location of batch.
NTO	(I*4)	The WASTES ID of the destination facility.
CTYPE	(I*4)	ID of cask used for the shipment.

WRTWRK Subroutine WRTWRT writes the contents of the WRKBAT array to unit NPRNT (FOR006 - .PRT).

WRTWRK

XFER Subroutine XFER effects the transfer of individual batches of fuel from the source queue to the destination queue.

XFER (NTO ,ISTOR , NASMS , BATAMT, LSTCSK, CTYPE)

NTO	(I*4)	The WASTES ID of the destination facility.
ISTOR	(I*4)	The WASTES offset ID of the current physical location of the batch. (if located in a pool of shared pair, the ID is the lower ID of the shared pair).
NASMS	(I*4)	The number of assemblies in the batch.
BATAMT	(R*4)	Weight of the batch in MTU.
LSTCSK	(L*4)	If TRUE, indicates that the costs for this shipment should be calculated for a full cask even though the cask is not full.
CTYPE	(I*4)	ID of cask used for the shipment.

APPENDIX B

SUBROUTINE/FUNCTION CALLING SEQUENCES

APPENDIX B

SUBROUTINE/FUNCTION CALLING SEQUENCES

Module Name	Called From	Calls To
ACCEPT	FRCAPO FRCAPP FRCAPS TRNLST	
ADRCST	FAC	SETVAL
AGECAT (FXN)	FCINVC MOV2DRY RXINVC UPDISA XFER	
BANNER	PROGRAM WASTES	DISCLAIM
BIOFMT	FAC	SETVAL COPY CPYFROM FETCSK SPLIT XFER RMOVE
BLOKDAT	N/A	N/A
BOOBOO	INTLC	
CHGALT	EVENT	GETAA ERROR EXPAND
CHGCAP	EVENT	ERROR GETAA EXPAND
CHGPRI	EVENT	GETAA ERROR EXPAND
CHKDRY	FAC	SETVAL COPY CPYFROM OTPUT

Module Name	Called From	Calls To
CHKINT	FAC	FETCSK
COSTER	OTPUT	DRYCST
CPYFROM	BIOFMT	XXERR
	CHKDRY	
	DCMFIL	
	DECOM	
	DSBCH	
	DUMPQ	
	FCINVC	
	FILLCSK	
	GETLOW	
	HTCHAR	
	MOV2DRY	
	MV2POOL	
	RXDIS	
	RXINVC	
	TRNLST	
	TRNSHP	
	UPDECOM	
CPYTO		XXERR
	DECOM	
	DSBCH	
	MOV2DRY	
	MV2POOL	
	SPLIT	
	XFER	
CSKUSE (FXN)	INTLC	
CVIC	INTLC	
DCMFIL	DECOM	COPY
		CPYFROM
		SPLIT
DECOM	EVENT	SETVAL
		ULINK
		LINK
		MV2POOL
		COPY
		CPYFROM
		FETCSK
		ROUND
		SPLIT
		UPDSRC
		XFER
		DCMFIL
		REMOVE

Module Name	Called From	Calls To
		CPYTO FILEM FACORDR SCHDL
DISCALC	INTLC	
DISCLAIM	BANNER	
DRYCST	COSTER	
DSBCH	RXDIS	FETCSK REMOVE FILEM XFER CPYTO PUTBAT CPYFROM
DUMPQ	INTERACTIVE DEBUGGER	COPY CPYFROM WRTWRK
ECHO	INTLC	
EVENT	SLAM	RXDIS FAC CHGCAP CHGPRI DECOM TRNSHP CHGALT
EXPAND	CHGALT CHGCAP CHGPRI FRCAPO FRCAPP FRCAPS	
EXPCAT (FXN)	FCINVC RXINVC XFER	
EXPRXID	INTLC	
FAC	EVENT	SCHDL UPDHT FRCAPO FRCAPS

Module Name	Called From	Calls To
		FRCAPP BIOFMT ADRCST ERROR UPDECOM HTCHAR SETVAL CHKDRY CHKINT RXSHPC FCSHPC RXINVC FCINVC
FACORDR	DECOM FRCAPP RXDIS	SETVAL SHELL
FCINVC	FAC	COPY CPYFROM
FCSHPC	FAC	
FETCSK	BIOFMT CHKINT DECOM DSBCH FRCAPO FRCAPP FRCAPS OPTFUNC RXDIS TRNSHP	
FILLCSK	FRCAPO FRCAPP FRCAPS	COPY CPYFROM SPLIT
FNDFST (FXN)	DECOM	COPY
FRCAPO	FAC	EXPAND GETLOW ACCEPT FETCSK ROUND SPLIT UPDSRC REMOVE FILEM XFER FILLCSK

Module Name	Called From	Calls To
		MV2POOL
FRCAPP	FAC	EXPAND FETCSK GETLOW FACORDR ROUND ACCEPT TRNLST SPLIT UPDSRC XFER WRTWRK FILLCSK MV2POOL
FRCAPS	FAC	EXPAND FETCSK GETLOW ROUND ACCEPT TRNLST SPLIT UPDSRC XFER WRTWRK FILLCSK MV2POOL
GENTRP (FXN)	HFUNC	
GETLOW	FRCAPO FRCAPP FRCAPS	COPY CPYFROM
GREATC (FXN)	DISCALC UPDISA	
HFUNC (FXN)	FCINVC HTCHAR MOV2DRY RXINVC SCHDIS UPDHT XFER	
HRANGE (FXN)	FCINVC HTCHAR MOV2DRY RXINVC XFER	

Module Name	Called From	Calls To
IINV (FXN)	CHGCAP DECOM DSBCH FAC FRCAPO FRCAPP FRCAPS MOV2DRY MV2POOL UPDISA XFER	
INTLC	SLAM	IDATE* TIME* SETAA SETVAL ECHO CVIC BOO BOO SETORDR EXPRXID SETRNG VERIFY RDPLNT SCHDIS SHELLMSM SETUP SCHDL SHELLM PROEVNT DISCALC
LEVEL	SPFOUT	SETVAL
MOV2DRY	RXDIS TRNSHP	COPY CPYFROM SPLIT REMOVE CPYTO FILEM
MRKTIM	INTERACTIVE DEBUGGER	
MV2POOL	DECOM FRCAPO FRCAPP FRCAPS	COPY CPYFROM SPLIT REMOVE CPYTO FILEM

Module Name	Called From	Calls To
OPTFUNC (FXN)	DECOM DSBCH FRCAPO	FETCSK SHPCST
OTPUT	CHKDRY RXDIS SLAM	PRXSHP SETVAL WRTMRS SPFOUT COSTER
PROEVNT	INTLC	SETVAL PUTBAT PUTAA SCHDL
PRXSHP	OTPUT	
PUTBAT	DSBCH PROEVNT SCHDIS SPLIT	XXERR
RDPLNT	INTLC	SETVAL FILEM SCHDL RMOVE
REMOVE	DECOM DSBCH FRCAPO MOV2DRY MV2POOL XFER	ULINK
ROUND	DECOM FRCAPO FRCAPP FRCAPS	
RXDIS	EVENT	SETVAL CPYFROM WRTWRK ERROR XFER OTPUT COPY CPYFROM SPLIT FETCSK DSBCH FACORDR

Module Name	Called From	Calls To
		TRNSHP MOV2DRY SCHDIS
RXINVC	FAC	COPY CPYFROM
RXSHPC	FAC	
SCHDIS	INTLC RXDIS	PUTBAT SCHDL
SETORDR	INTLC	
SETRNG	INTLC	
SETUP	INTLC	
SETVAL	ADRCST BIOFMT CHKDRY COSTER DECOM FAC FACORDR HTCHOT INTLC LEVEL OTPUT PROEVNT RDPLNT RXDIS SPFOUT UPDECOM	
SHELL	FACORDR	
SHELLM	INTLC	
SHELLMSM	INTLC	
SHPCST	OPTFUNC UPDISA	
SLAM	PROGRAM WASTES	INTLC EVENT OTPUT

Module Name	Called From	Calls To
SPFOUT	OTPUT	SETVAL ERROR LEVEL
SPLIT	BIOFMT DCMFIL DECOM FILLCSK FRCAPO FRCAPP FRCAPS MOV2DRY MV2POOL RXDIS TRNLST TRNSHP	WRTWRK ERROR CPYTO PUTBAT FILEM
TRNLST	FRCAPP FRCAPS	ACCEPT FILEM COPY CPYFROM RMOVE SPLIT XFER ULINK
TRNSHP	RXDIS	COPY ULINK FETCSK SPLIT XFER CPYFROM MOV2DRY
UPDECOM	FAC	SETVAL COPY CPYFROM
UPDHT	FAC	
UPDISA	XFER	SHPCST
UPDSRC	DECOM FRCAPO FRCAPP FRCAPS	
VERIFY	INTLC	

Module Name	Called From	Calls To
WBAT (FXN)	DECOM FNDFST MV2POOL	ERROR
WRTMRS	OTPUT	
WRTSHP	XFER	
WRTWRK	DUMPQ FRCAPP FRCAPS RXDIS SPLIT	
XFER	BIOFMT DECOM DSBCH FRCAPO FRCAPP FRCAPS RXDIS TRNLST TRNSHP	ULINK LINK WRTSHP COPY CPYTO FILEM REMOVE UPDISA

APPENDIX C

INCLUDED COMMON BLOCK DESCRIPTIONS

APPENDIX C

INCLUDED COMMON BLOCK DESCRIPTIONS

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
BANNER	WASTES	BLOCKD	VERSION	C*11
			RELEASE	C*11
			TYPEX	C*11
Included In: BLOKDAT, INTLC, SPFOUT, PROGRAM WASTES				
BAT	WASTES	BAT	MFBAT	I*4
			WRKBAT(NWRKA)	R*4
			BATCH(NDBAT)	R*4
			DIMENSION IBATCH(NDBAT)	I*4
			DIMENSION IWRKBAT(NWRKA)	I*4
			EQUIVALENCE (IWRKBAT, WRKBAT)	
			EQUIVALENCE (IBATCH, BATCH)	
		EVN	MFEVN	I*4
			EVENT(NDEVN)	R*4
Included In: ACCEPT, ADRCST, BIOFMT, BLOKDAT, CHGALT, CHGCAP, CHGPRI, CHKDRY, CPYFROM, CPYTO, DCMFIL, DECOM, DRYCST, DSBCH, DUMPQ, FAC, FCINVC, FILLCSK, FNDFST, FRCAPO, FRCAPP, FRCAPS, HTCHAR, INTLC, MOV2DRY, MV2POOL, OTPUT, PROEVNT, PUTBAT, RDPLNT, REMOVE, RXDIS, RXINVC, SCHDIS, SPFOUT, SPLIT, TRNLST, TRNSHP, UPDECOM, UPDHT, UPDISA, WBAT, WRTSHP, WRTWRK, XFER				
CASK	WASTES	CASK	EMPTWT(MAXCASKS)	R*4
			CSKRNT(MAXCASKS)	R*4
			CSKCST(MAXCASKS)	R*4
			CSKDYS(MAXCASKS)	R*4
			TRNRND(MAXCASKS)	R*4
			CSKOAM(MAXCASKS)	R*4
			CSKLIF(MAXCASKS)	R*4
			TRKONLY	L*4
			TRNONLY	L*4
			LEASE	L*4
			LAGPOOL	R*4
			TFACTR	R*4
			RFACTR	R*4
			MCASK(10)	C*8
			NMSHIP(10)	I*4
			DOLMIL(10)	R*4
			SMULT(10)	R*4
			TURNRD(10)	R*4
			LSTTRN	L*4

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
Included In: BLOKDAT, DECOM, FRCAPP, FRCAPS, INTLC, RDPLNT, RXDIS, SHPCST, SPFOUT, TRNLST, UPDISA				
CTCOM	WASTES	CTCOM	FCINV(4, MXFAC2)	R*4
			FCSHIP(4, MXFAC2, MXFAC2)	R*4
			CCSTF(4, MXFAC2, MXFAC2)	R*4
			SCCST(4, MXFAC2, MXFAC2)	R*4
			TCTONS(2, MXFAC2)	R*4
			ZCTONS(2, MXFAC2)	R*4
Included In: ADRCST, BLOKDAT				
DATES	WASTES	DATES	QQMON	I*4
			QQDAY	I*4
			QQYEAR	I*4
			QQTIME	C*8
Included In: BLOKDAT, BOOBOO, INTLC, OTPUT, SPFOUT				
DECOM	WASTES	RPTDEC	DECLAG	R*4
			DCAGE(6)	R*4
			TYMDEC(NREAC)	R*4
			LAG	R*4
			AGE1	R*4
			AGE2	R*4
			AGE3	R*4
			DCMSHP(2)	R*4
			IDSHP(NREAC)	I*4
Included In: BLOKDAT, CHKINT, INTLC, OTPUT, RDPLNT, UPDECOM, XFER				
DEFAULT	WASTES	DEFAULTC	FHDDEF	C*20
			DEFDAT(4,2)	C*50
			CSKTYP(3)	C*5
			DSTTRK	C*50
			DSTRAL	C*50
		DEFAULTN	CSKDEF(9,3)	R*4
			AGEDEF(6,2)	R*4
			FSDEF	R*4
			FLATDEF	R*4
			FLNGDEF	R*4
			TRKDEF	L*4
			TRNDEF	L*4
			LEASEDEF	L*4
			RXRNG1	I*4
			RXRNG2	I*4
			NINRNG	I*4
			DECOMDF	R*4
			TYPDEF	I*4

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
			DRTYPDEF	I*4
			IDCNTYR	I*4
			DCNTDEF	R*4
			FLATL	R*4
			FLATU	R*4
			FLONGL	R*4
			FLONGU	R*4

Included In: BLOKDAT, INTLC

DISCOM	WASTES	DISCOM	TTTBEG	R*4
			FSDATE	R*4
			FSMTU	R*4
			FSTOPD	L*4

Included In: BLOKDAT, CHKINT, FRCAPO, FRCAPP, FRCAPS, INTLC, SCHDIS

DRYTYP	WASTES	DRYTYP	DTYP (MAXFILS)	I*4
			DRTYP	I*4
			NAPC (MXDFAC)	I*4
			DIMENSION NUMAPC(2, MXDFAC/2)	I*4
			EQUIVALENCE (NAPC(1), NUMAPC(1,1))	

Included In: ADRCST, BIOFMT, BLOKDAT, CHKDRY, DECOM, DRYCST, DSBCH, FRCAPO, FRCAPP, FRCAPS, INTLC, MOV2DRY, RDPLNT, RXDIS, UPDISA

GCOM1	SLAM	GCOM1	JJCDR	I*4
			KKNN	I*4
			LLFIL	I*4
			LLRNK	I*4
			LLTRY	I*4
			MFEX	I*4
			NNAM1	I*4
			NNAM2	I*4
			NNAM3	I*4
			NNAPO	I*4
			NNAPT	I*4
			NNATR	I*4
			NNFIL	I*4
			NNTRY	I*4
			TTBEG	R*4
			TTCLR	R*4
			TTFIN	R*4
			TTSET	R*4
			XXI(100)	R*4
			TTTS	R*4
			TTTF	R*4

Included In: BLOKDAT, DRYCST, DSBCH, FAC, FRCAPO, INTLC, OTPUT, RDPLNT, REMOVE, SPFOUT, UPDISA, XFER

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
GCOM4	SLAM	GCOM4	DTPLT(10)	R*4
			HHLOW(50)	R*4
			HHWID(50)	R*4
			IICRD	I*4
			IITAP(10)	I*4
			JJCEL(500)	I*4
			LLABC(50,4)	I*4
			LLABP(11,4)	I*4
			LLABT(50,4)	I*4
			LLPHI(10)	I*4
			LLPLO(10)	I*4
			LLPLT	I*4
			IIPLT(10)	I*4
			IIVPT(10,10)	I*4
			LLSUP(2)	I*4
			LLSYM(10)	I*4
			MMPTS	I*4
			NNCEL(50)	I*4
			NNCLT	I*4
			NNHIS	I*4
			NNPLT	I*4
			NNTMX	I*4
			NNTMA	I*4
			NNPTS(10)	I*4
			NNSTA	I*4
			NNVAR(10)	I*4
			PPHI(10)	R*4
			PPLO(10)	R*4
			IITMX(50)	I*4
			IITMA(50)	I*4
			NNHC(50)	I*4
			NNCH(50)	I*4
			KKSUP(10)	I*4
			TTSRT(10)	R*4
			TTEND(10)	R*4
			JJPLT(10)	I*4
Included In: BLOKDAT, INTLC				
GCOM5	SLAM	GCOM	IISED(10)	I*4
			JJBEG	I*4
			JJCLR	I*4
			MMNIT	I*4
			MMON	I*4
			NNAME(5)	I*4
			NNCFI	I*4
			NNDAY	I*4
			NNPT	I*4
			NNPRJ(5)	I*4
			NNRNS	I*4

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
			NNSTR	I*4
			NNYR	I*4
			SSEED(10)	R*4
			LSEED(10)	I*4
Included In: BLOKDAT, BOOBOO, INTLC, OTPUT, SPFOUT				
GCOM6	SLAM	GCOM6	EENQ(651)	R*4
			IINN(651)	I*4
			KKRNK(651)	I*4
			MMAXQ(651)	I*4
			QQTIM(651)	R*4
			SSOBV(50,5)	R*4
			SSTPV(50,6)	R*4
			VVNQ(651)	R*4
Included In: BLOKDAT, INTLC, SETORDR				
GCOM7	SLAM	GCOM7	IIECH	I*4
			IIFIN	I*4
			IINNA(4)	I*4
			IIPOF	I*4
			IIPOS	I*4
			IISUM	I*4
			IIPIR	I*4
			IERRF	I*4
			IRROR	I*4
			IVART(4)	I*4
			JJFIL	I*4
			JBEGG	I*4
			JLST	I*4
			KARD(80)	I*4
			KREAD(20)	I*4
			LLINE	I*4
			MORE	I*4
			NEXT	I*4
			NFLD	I*4
			NRTOT	I*4
			NTRM(50)	I*4
			VALUE(50)	R*4
			IP	I*4
			NQXCD	I*4
			IIMOD	I*4
Included In: BLOKDAT, INTLC, RDPLNT				

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
HETCOM	WASTES	HETCOM	CAT(NCAT)	R*4
			HEAT(NCAT)	R*4
			CATAGE(NCAT)	R*4
			CATEXP(NECAT)	R*4
Included In:	AGECAT, BLOKDAT, EXPCAT, FAC, FRCAPO, FRCAPP, FRCAPS, HRANGE, HTCHAR, HTCHOT, INTLC, OTPUT, RDPLNT, SPFOUT, TRNSHP, UPDISA, XFER			
MRSCST	WASTES	ALONE	TRUCKI(1962:1962+NBRYR)	R*4
			TRAINI(1962:1962+NBRYR)	R*4
			PWRS(1962:1962+NBRYR,3)	I*4
			BWRS(1962:1962+NBRYR,3)	I*4
			PWRR(1962:1962+NBRYR,3)	I*4
			BWRR(1962:1962+NBRYR,3)	I*4
			PWRI(1962:1962+NBRYR,3)	I*4
			BWRI(1962:1962+NBRYR,3)	I*4
Included In:	BLOKDAT, WRTMRS, XFER			
NDRYCOM	WASTES	DRYCOM	CAPYRD(MXDFAC,NBRYR)	R*4
			CAPCSK(MXDFAC,NBRYR)	R*4
			OMYRD(MXDFAC,NBRYR)	R*4
			OMCSK(MXDFAC,NBRYR)	R*4
			DCYRD(MXDFAC,NBRYR)	R*4
			DCCSK(MXDFAC,NBRYR)	R*4
			SALVAG(MXDFAC,NBRYR)	R*4
			NCASKS(MXDFAC,NBRYR)	I*4
			NYARDS(MXDFAC,NBRYR)	I*4
			NASSM(MXDFAC,NBRYR)	I*4
			NYEAR(NBRYR)	I*4
			DDOLRS(MAXFILS,2)	R*4
			DASMCT(MAXFILS,2)	R*4
Included In:	ADRCST, BLOKDAT, DRYCST, OTPUT			
OTCOM	WASTES	'BLANK COMMON'	DIMENSION QSET(NDIMQSET)	R*4
			EQUIVALENCE (QSET(1),FACINV(1,1,1))	
			FACINV(2,NBRYR,MXFACID)	R*4
			SHIP(2,MXFACID,NBRYR,MXFACID)	R*4
			COST(NBRYR,NUMCAT,MXFACID)	R*4
			FPARAM(2,3)	I*4
Included In:	BLOKDAT, COSTER, DRYCST, OTPUT, SPFOUT			
OTONLY	WASTES	OTONLY	RXS(9,NREAC,MXFACID)	R*4
			RXSASM(MXFACID)	R*4
			RXSMTS(MXFACID)	R*4

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
Included In: BLOKDAT, INTLC, SPFOUT, UPDISA				
OTP	WASTES	INVOUT	RXAGE (7,1962:1962+NBRYR)	R*4
			RXHEAT (7,1962:1962+NBRYR)	R*4
			RXEXPOS (6,1962:1962+NBRYR)	R*4
			RXASTS (3,1962:1962+NBRYR)	R*4
			RXESTS (3,1962:1962+NBRYR)	R*4
			RXHSTS (3,1962:1962+NBRYR)	R*4
			RXIASM (1962:1962+NBRYR)	I*4
		SHPOUT	RSAGE (7,1962:1962+NBRYR)	R*4
			RSHEAT (7,1962:1962+NBRYR)	R*4
			RSEXPOS (6,1962:1962+NBRYR)	R*4
			RSASTS (3,1962:1962+NBRYR)	R*4
			RSESTS (3,1962:1962+NBRYR)	R*4
			RSHSTS (3,1962:1962+NBRYR)	R*4
			RXOASM (1962:1962+NBRYR)	I*4
			RSAMIN	R*4
			RSEMIN	R*4
			RSHMIN	R*4
			RSAMAX	R*4
			RSEMAX	R*4
			RSHMAX	R*4
			RSAAVG	R*4
			RSEAVG	R*4
			RSHAVG	R*4
		FACOUT	FCAGE (7,MXFACID,1962:1962+NBRYR)	R*4
			FCHEAT (7,MXFACID,1962:1962+NBRYR)	R*4
			FCEXPOS (6,MXFACID,1962:1962+NBRYR)	R*4
			FCASTS (3,MXFACID,1962:1962+NBRYR)	R*4
			FCESTS (3,MXFACID,1962:1962+NBRYR)	R*4
			FCHSTS (3,MXFACID,1962:1962+NBRYR)	R*4
FCIASM (MXFACID,1962:1962+NBRYR)			I*4	
		FACIN	FIAGE (7,MXFACID,MXFACID, 1962:1962+NBRYR)	R*4
			FIHEAT(7,MXFACID,MXFACID, 1962:1962+NBRYR)	R*4
			FIEXP (6,MXFACID,MXFACID, 1962:1962+NBRYR)	R*4
			FIASTS (3,MXFACID,MXFACID, 1962:1962+NBRYR)	R*4
			FIHSTS (3,MXFACID,MXFACID, 1962:1962+NBRYR)	R*4
			FIESTS (3,MXFACID,MXFACID, 1962:1962+NBRYR)	R*4
			FIASMS (MXFACID,MXFACID, 1962:1962+NBRYR)	I*4

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
			FCMIN (3,MXFACID,MXFACID)	R*4
			FCAVG (3,MXFACID,MXFACID)	R*4
			FCMAX (3,MXFACID,MXFACID)	R*4
Included In: FCINVC, FCSHPC, INTLC, OTPUT, RXINVC, RXSHPC, XFER				
RCOM	WASTES	RCOM	RTYPE (NREAC)	I*4
			NERC (NREAC)	I*4
			LAT (NREAC)	R*4
			LONG (NREAC)	R*4
			RAIL (NREAC)	I*4
			TTYP (NREAC)	I*4
			CENTRD (NREAC)	I*4
			RUNIDS (NREAC)	I*4
			FCR (NREAC)	R*4
			NRUNIDS (NREAC)	I*4
			RXCASKS (MXRXCSK,NREAC)	I*4
			FCRFRAC	R*4
			ORGFCR (NREAC)	R*4
			UTILID (NREAC)	I*4
			NTRANS	I*4
			PREVENT	L*4
Included In: ADRCST, BIOFMT, BLOKDAT, CHGCAP, CHKINT, DCMFIL, DCMFIL, DECOM, DISCALC, DRYCST, DSBCH, EVENT, FAC, FACORDR, FETCST, FILLCSK, FRCAPO, FRCAPP, FRCAPS, HTCHAR, INTLC, MOV2DRY, MV2POOL, OPTFUNC, PROEVNT, PRXSHP, RDPLNT, RXDIS, SCHDIS, TRNSHP, UPDHT, UPDISA, XFER				
SCOM1	SLAM	SCOM1	ATRIB(100)	R*4
			DD(100)	R*4
			DDL(100)	R*4
			DTNOW	R*4
			II	I*4
			MFA	I*4
			MSTOP	I*4
			NCLNR	I*4
			NCRDR	I*4
			NPRNT	I*4
			NNRUN	I*4
			NNSET	I*4
			NTAPE	I*4
			SS(100)	R*4
			SSL(100)	R*4
			TNEXT	R*4
			TNOW	R*4
			XX(100)	R*4
			DIMENSION IATRIB(100)	I*4
			EQUIVALENCE (IATRIB, ATRIB)	

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
Included In:	ACCEPT, ADRCST, BIOFMT, BLOKDAT, BOOBOO, CHGALT, CHGCAP, CHGPRI, CHKDRY, CHKINT, COSTER, CPYFROM, DCMFIL, DECOM, DISCALC, DRYCST, DSBCH, DUMPQ, EVENT, EXPAND, FAC, FCINVC, FCSHPC, FILLCSK, FISCST, FNDFST, FRCAPO, FRCAPP, FRCAPS, GETLOW, HTCHAR, HTCHOT, HTFILE, INTLC, MOV2DRY, MRKTIM, MV2POOL, OTPUT, PROEVNT, PRXSHP, PUTBAT, RDPLNT, REMOVE, RXDIS, RXINVC, RXSHPC, SCHDIS, SPFOUT, SPLIT, TRNLST, TRNSHP, UPDECOM, UPDHT, UPDISA, PROGRAM WASTES, WBAT, WRTSHP, WRTWRK, XFER			
UCOM	WASTES	UCOM	DBGDRY	L*4
			DBGINT	L*4
			DRYERR	L*4
			INTERR	L*4
			MOROTP	L*4
			NREACTORS	I*4
			VCSKTM	R*4
			CSKTIM(NREAC)	R*4
			DRYTRN	R*4
			CSHPMT(MAXCASKS,MXFACID)	R*4
			REPRO	R*4
			MRS	R*4
			REPOS	R*4
			FIS	R*4
			DRY	R*4
			POOL	R*4
			IDREPRO	I*4
			IDMRS	I*4
			IDREPOS	I*4
			IDFIS	I*4
			IDDRY	I*4
			IDPOOL	I*4
			ISTART	I*4
			IFIRSTR	I*4
			ILASTR	I*4
			NFAC	I*4
			NPOOL	I*4
			NDRY	I*4
			ALOWHT	L*4
			MAXID	I*4
			INVFIL	I*4
			CSKFIL	I*4
			HTFIL	I*4
			RDTODT	R*4
			NUMFAC(NUMFAX)	I*4
			FACBAS(NUMFAX)	I*4
			IFAC(2,MXFACID)	I*4
			CAP(MAXFILS)	R*4
			CAPMAX(MAXFILS)	R*4
			IPOOL(MAXFILS)	I*4

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
			PREVDRY(MAXFILS)	R*4
			CURDRY(MAXFILS)	R*4
			FDATE(MAXFILS)	R*4
			PRIOR(PMAX,MXFACID)	R*4
			IPUNL(MXFACID)	I*4
			ALTPRI(MXFACID,MXFACID)	I*4
			AGE(4,MXFACID)	R*4
			FINV(2,MXFACID)	R*4
			FSHIP(2,MXFACID,MXFACID)	R*4
			UNLOAD(MXFACID)	R*4
			UNLMAX(MXFACID)	R*4
			ISTIME	I*4
			SPRMAX(MXFACID)	R*4
			SPRINT(MXFACID)	R*4
			COSTF(4,MXFACID,MXFACID)	R*4
			HTINP(MXFACID)	L*4
			HTMAX(MXFACID)	R*4
			FACCAP(4)	R*4
			FYRCAP(4)	R*4
			IREACT(10)	I*4
			CSTFLG	L*4
			IBYR	I*4
			DISCNT	R*4
			FTYPE(4)	R*4
			FLGREPR	R*4
			FLGMRS	R*4
			FLGREPO	R*4
			FLGFIS	R*4
			FILRPRO	I*4
			FILMRS	I*4
			FILFIS	I*4
			GTOTASMS	R*4
			GTOTMTS	R*4
			FACFLG(4)	I*4
			FACSOPT(2,4)	I*4
			FCRPRI(MXFACID-2)	R*4
			DCMPRI(MXFACID-2)	R*4
			CASMD(MAXFILS)	I*4
			PASMD(MAXFILS)	I*4
			FACOPT(4)	L*4
			DIMENSION OPTFAC(4)	R*4
			DIMENSION FACFIL(4)	I*4
			EQUIVALENCE (OPTFAC(1), FLGREPR)	
			EQUIVALENCE (FACFIL(1), FILRPRO)	
		TCOM	AMTS(15,2,MXFACID)	R*4
			SCASKS(2,MAXCASKS,MXFACID)	R*4
			PCPCASK(MAXWASTE,MAXCASKS)	I*4
			SCOSTS(4,MAXCASKS,MXFACID)	R*4

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
			FACLTY(2,MXFACID)	R*4
			SUMHT(7,MXFACID)	R*4
			TCASKS(3,MXFACID)	R*4
			TTONS(MAXCASKS+1,MXFACID)	R*4
			FACSTA(8,2,MXFACID)	R*4
			OCASKS(2,2,MXFACID)	R*4
			IASSM(7,2,MXFACID)	I*4
			JASSM(7,2,MXFACID)	I*4
			TBLHD(MXFACID)	C*17
			FACTTL(MXFACID)	C*17
			FUELC(3,2,21,MXFACID)	R*4
			ZTONS(MAXCASKS,MXFACID)	R*4
			FACCM(2,2,21,MXFACID)	R*4
			DISFDG(2)	R*4
			SRC2TRG(MXFACID-2,NREAC+MXFACID-2)	R*4
			SHPCSK(2,MAXCASKS)	C*8
			FACCSK(MAXCASKS,MXFACID,2)	I*4
			NUMCSK	I*4
			NUMWSTYP	I*4
			SHEAD1	C*32
			SHEAD2	C*24
			CSKMIL(MAXCASKS,MXFACID, 1960:1960+NBRYR)	R*4
Included In:	ACCEPT, ADRCST, BIOFMT, BLOKDAT, CHGALT, CHGCAP, CHGPRI, CHKDRY, CHKINT, COSTER, DCMFIL, DECOM, DISCALC, DRYCST, DSBCH, DUMPQ, EVENT, EXPAND, EXPRXID, FAC, FACORDR, FCINVC, FETCSK, FILLCSK, FNDFST, FRCAPO, FRCAPP, FRCAPS, GETLOW, HTCHAR, HTCHOT, HTFILE, IINV, INTLC, MOV2DRY, MV2POOL, OPTFUNC, OTPUT, PROEVNT, PRXSHP, PUTBAT, RDPLNT, REMOVE, RXDIS, RXINVC, SCHDIS, SETUP, SHPCST, SPFOUT, SPLIT, TRNLST, TRNSHP, UPDECOM, UPDISA, UPDSRC, WRTSHP, XFER			
UCOM2	WASTES	UCOM2	FCOUNT(NREAC)	I*4
			FREFILE(NREAC)	I*4
			TREORDR(NREAC)	R*4
			DLTATIM	R*4
			TPREV	R*4
Included In:	BLOKDAT, CHGCAP			
XCOM1	SLAM	XCOM1	MMFE(651)	I*4
			MMLE(651)	I*4
			NQ(651)	I*4
Included In:	BLOKDAT, OTPUT, SPFOUT			

Included Name	WASTES/SLAM	Name of Common	Arrays/Variables	Type
XCOM5	SLAM	XCOM5	NECLT	I*4
			IIERP	I*4
			IIXQT	I*4

Included In: BLOKDAT, INTLC

APPENDIX D

SLAM QUEUES USED BY WASTES II

APPENDIX D

SLAM QUEUES USED BY WASTES II

As previously mentioned, WASTES uses dynamically offset queue IDs for facilities (both reactors and destination facilities). A number of SLAM queues are also used by WASTES as working queues and are listed below in tabular form. Each waste management/processing facility is assigned two queues according to the order in which the facilities were nominated on the WASTES input cards.

For example, if an MRS facility, a Repository 1, and a Repository 2, are specified in that order in the WASTES input cards, the WASTES facility IDs will be, respectively, 1, 2, and 3. Following the waste management facility queue assign-ments, the generic pool (NPOOL) and generic dry (NDRY) facility IDs will be made sequentially, in this case 4 and 5 respectively.

To obtain the time sorted queue IDs, the WASTES facility IDs are multiplied times 2. The formula to obtain the ID of the queues reserved for future expansion is 2 * ID - 1. The queue assignments resulting from this example are shown below.

Facility	Queue	Queue Use
MRS	1	Reserved for Future Expansion
MRS	2	Time Sorted Queue
REPOSITORY 1	3	Reserved for Future Expansion
REPOSITORY 1	4	Time Sorted Queue
REPOSITORY 2	5	Reserved for future Expansion
REPOSITORY 2	6	Time Sorted Queue
NPOOL	7	Reserved for Future Expansion
NPOOL	8	Time Sorted Queue
NDRY	9	Reserved for Future Expansion
NDR	10	Time Sorted Queue

The number of queues that are used for waste management facilities and the generic queues are represented by the variable ISTART. In the case of the above mentioned example, ISTART equals 10.

At this point the reactor queue assignments begin and another offset may be introduced. This additional offset is IFIRSTR, which contains the ID of the first reactor to be included in the simulation. If the first reactor read in from the reactor site data file is not the first reactor to be included in the simulation, the number of SLAM queues required for the simulation to run will be reduced by the value of IFIRSTR.

Reactor queue assignments are then made according to the following formula reactor queue = reactor ID + 1 + ISTART - IFIRSTR. This sequence continues for each reactor as it is read in from the reactor site data file. As the reactor IDs read in from this file are not monotonically increasing, the SLAM queues will not be monotonically increasing either. This will result in some unused SLAM queues.

The other SLAM queues used during a WASTES run as working queues are listed below.

Queue Use	Queue ID	Queue User
Event Calendar	MFIL + 1	SLAM
Transshipment	MFIL	WASTES
Decommissioning Reactors	MFIL - 1	WASTES
Optimized Queue for Reprocessing	MFIL - 2	WASTES
Optimized Queue for MRS	MFIL - 3	WASTES
Optimized Queue for Repositories	MFIL - 4	WASTES
Optimized Queue for FIS	MFIL - 5	WASTES
Multi-Cask Rail Shipments	MAX OFFSET REACTOR ID+5	WASTES

Where: MFIL is a SLAM variable read in from the limits card and represents the largest file number.

APPENDIX E

FORTRAN UNITS USED BY WASTES II

APPENDIX E

FORTRAN UNITS USED BY WASTES II

Logical Unit Number	Assignment
1	Post Processor Data File
2	Scratch File - WASTES Input Cards
3	Scratch File
5	Input File
6	Output File
7	Scratch File
9	Centroid Data File
10	Discharge Data File
11	Reactor Data File
15	Scratch File - Annual Inventories
16	Scratch File - Annual Shipments
.	.
.	.
27	Scratch File - Annual Shipments
30	Scratch File - Info. Aggregated by Casks
.	.
.	.
42	Scratch File - Info. Aggregated by Casks
45	Scratch File - Heat Information
.	.
.	.
59	Scratch File - Heat Information
70	Bi-Annual Reactor Shipments (assemblies)
80	Dry Storage Inventory
81	Facility 1 Heat/Age Inventory
82	Facility 2 Heat/Age Inventory
83	Facility 3 Heat/Age Inventory
.	.

.	.
.	.
90	Facility 10 Heat/Age Inventory
95	Annual Reactor Shipments (assemblies)
96	Bi-Annual Reactor Shipments (mtu)
97	Scratch File - Decommissioned Reactor Information
98	File for MRS cost model
99	Shipment Log File for SIMAN model

FORTRAN units not listed above are currently unused.

APPENDIX F

MAJOR VARIABLE DESCRIPTIONS

APPENDIX F

MAJOR VARIABLE DESCRIPTIONS

The following list of major variables includes those variables that are widely used throughout WASTES or whose usage is not immediately obvious. Local variables often have different meanings between subroutines and may have different meanings within the same subroutine. This listing makes no attempt to be all inclusive. In general, local variable usage will have to be determined from the context of the subroutine which it is used in.

BATAMT	The number of MTU in a batch or sub-batch.
CSKASMS	The number of assemblies in a cask. The cask may be either a shipping cask or a dry storage cask. The value may change within subroutine.
IBASE	The WASTES pointer to the first facility (lowest minor ID) of a facility type.
IFIRSTR	The real reactor ID of the first reactor to be included in the simulation.
ISTART	The variable WASTES offset for SLAM queue IDs. Reactor IDs begin at ISTART + 1. ISTART = 2 * # of facilities being modelled + 4. See Appendix D for more information queue offsets.
ISTOP	A logical flag used to stop execution of WASTES if fatal errors are detected in subroutine INTLC.
JBASE	The WASTES pointer to the last facility (highest minor ID) of a facility type.
LSTCSK	If TRUE, indicates that the costs for this shipment should be calculated for a full cask even though the cask is not full. This is used in the case of the last cask leaving a reactor that is being decommissioned, where the cask may not be full.
LSTTRN	When the last cask in a train is shipped, (either full or partial) the LSTTRN flag is set true. This flag indicates to the costing routines that the costs applicable only to multi-cask rail shipments should be applied in addition to regular transportation costs (shipping).

MARKED	MARKED is a logical flag which, if true, indicates to subroutine RXDIS that a partial cask shipment has occurred and that more material must be shipped in order to ship integer shipping cask quantities. MARKED is set in subroutine DSBCH.
NDRY	NDRY holds the WASTES pointer to arrays holding values that represent generic dry storage.
NASMS	NASMS contains the number of assemblies in a fuel batch or sub-batch.
NPOOL	NPOOL holds the WASTES pointer to arrays holding values that represent generic reactor pool storage.
NPRNT	The SLAM variable that contains the FORTRAN unit to be used for the print file. In the WASTES VAX/VMS implementation, NPRNT = 6.
NTRY	NTRY contains the pointer to the batch array.
TNOW	The SLAM variable that holds the current value of the simulated time.

APPENDIX G

MAJOR PARAMETER DESCRIPTIONS

APPENDIX G

MAJOR PARAMETER DESCRIPTIONS

PARAMETER NDIMQSET = 500000	Dimension of SLAM array QSET that holds attribute values and the associated pointers.
PARAMETER NBATA = 8	Number of fields in BATCH array into which WRKBAT values are compressed.
PARAMETER NDBAT = 300000	Maximum number of batches allowed in the simulation.
PARAMETER NWRKA = 12	Number of fields in WRKBAT array that is compressed into BATCH array.
PARAMETER I2AFLD = 1024	Value for decoding field 2.1 from the BATCH array.
PARAMETER I3AFLD = 2048	Value for decoding field 3.1 from the BATCH array.
PARAMETER I3BFLD = 64	Value for decoding field 3.2 from the BATCH array.
PARAMETER I3CFLD = 2	Value for decoding fields 3.3 and 3.4 from the BATCH array.
PARAMETER NEVN = 8	Number of auxiliary attributes associated with WASTES events.
PARAMETER NDEVN = 640	The dimension of SLAM's auxiliary filing array.
PARAMETER PMAX = 5	The number of WASTES priority queues.
PARAMETER MXFACID = 14	The maximum number of facility IDs including NPOOL and NDRY. Note: actual maximum number of facilities that can be modelled is 10 + NPOOL + NDRY.
PARAMETER MAXFILS = 651	The maximum number of files required. In practice this value must equal and has the same uses as NREAC.
PARAMETER NUMFAX = 6	Number of facility types (four classes of facilities + NPOOL + NDRY).

PARAMETER NCAT = 7	Number of categories into which age and heat characterizations are broken.
PARAMETER NECAT = 6	Number of categories into which exposure characterizations are broken.
PARAMETER NREAC = 651	The maximum number of reactors that can be modeled. This parameter also serves as an array dimension for many arrays and as a loop index in many instances. Note: the actual number of reactors which can be modelled is actually less than this value.
PARAMETER NBRYR = 100	The maximum number of years in the simulation. This parameter also serves as an array dimension for many arrays and as a loop index in many instances.
PARAMETER NUMCAT = 6	Not currently implemented.
PARAMETER MXDFAC = 12	The maximum number of dry storage costing categories. Currently costs are aggregated for each fuel type (PWR, BWR) within each of the six dry storage cask technologies.
PARAMETER MAXWASTE = 10	The maximum number of waste types. Note: The current maximum is two; PWR and BWR only.
PARAMETER MAXCASKS = 10	The maximum number of casks that may be modeled in the simulation.
PARAMETER MXRXCSK = 5	The maximum number of casks that may be assigned to any individual reactor.
PARAMETER IMAX = 2	*MXFACID*(MXFACID+1+NUMCAT/2)*(NBRYR). Not currently implemented.

APPENDIX H

GLOSSARY OF TERMS

APPENDIX H

GLOSSARY OF TERMS

This glossary contains terms germane to commercial waste management systems. Not all of the terms contained in this glossary are used in the WASTES II Programmers Reference Manual, however, all of these terms must be understood by the application programmer.

batch	a single fuel discharge from a reactor.
burnup	see exposure.
busy days	the number of days which a shipping cask must be used to complete a shipment.
BWR	boiling water reactor.
canister	a container which holds spent fuel rods or radioactive hardware which has been stripped off of the fuel rod assemblies.
cask	1. a shipping cask which contains spent fuel assemblies. The cask contains a specified number of PWR and BWR assemblies. The cask may be moved by truck or rail forms of transportation. 2. a dry storage cask which contains a specified number of PWR and BWR assemblies. These casks are generally not transportable. They are used to contain spent fuel at reactors which have insufficient room in the reactor pool to store the fuel and which could not ship the fuel at the time it was placed in dry storage.
consolidation	repackaging fuel assemblies so that they occupy less volume.
enrichment	the original concentration of U-235 in the fuel.
exposure	the amount of electricity produced by the fuel as measured in mega watt days/metric tonne. The exposure gives some indication of the time in the core and the resultant heat generation rate.
FCR	the full core reserve margin. As many empty storage locations which must remain in the pool as there are fuel assemblies in the reactor when it is fully loaded.
FIS	a Federal Interim Storage Facility. These facilities are similar to MRS facilities except they do not solicit fuel and are much more constrained as to application (by law).

fuel assemblies	a collection of fuel rods held together in a specific configuration. Typically 49 fuel rods in a PWR assembly and 196 in a BWR assembly.
fuel bundles	see fuel assemblies.
fuel pins	see fuel rods.
fuel rods	a single metal tube containing pellets of enriched uranium.
HLW	high level waste.
HTGR	high temperature gas reactor. The only operating commercial plant of this type in the U.S. is located at Ft. St. Vrain Colorado.
integer casking	1. shipping casks which are filled to capacity or empty. 2. dry storage casks which are filled to capacity or empty.
MRS	a Monitored Retrievable Storage facility. These facilities are the most general of the storage/processing facilities in that they receive, store, and ship spent fuel.
MTU	the metric tonnes of uranium contained in the fuel. This does not include the weight of the fuel rod cladding or other hardware.
NERC Region	North American Electric Reliability Council Regions. There are 9 regions in the U.S.
OCRWM	Office of Civilian Radioactive Waste Management.
pool	a water filled pool located in close proximity to the reactor core in which fuel assemblies (both new and used) are stored.
PWR	pressurized water reactor.
rack	metal racks located inside the reactor pool which hold the fuel assemblies in a vertical position.
repository	a geologic repository which represents the final disposal of nuclear waste in the waste management system.
reprocessing	a reprocessing plant extracts unused fissionable material from spent nuclear fuel. The plant then produces "new" fuel and a different type of radioactive waste.
rerack	when some reactor pools were constructed, they were either not completely racked or the have the possibility to be racked tighter. Reracking is the installation of new or additional racks to make the maximum utilization of the available pool storage capacity.

security & surcharges	extra costs involved in the shipment of fissionable materials.
skeletons	the fuel rod cladding after it has been removed from the fuel.
sub batch	a fuel batch which has been decreased in size (split).
transshipping	shipment of spent fuel between reactors pools. This allows reactors with small pools to take advantage of other reactors in the transshipment network which may have larger capacity pools.
TRU	transuranic waste.
turnaround time	the time required to complete the loading or unloading of a dry storage or shipping cask.

PNL-5728
UC-71

DISTRIBUTION

No. of Copies

OFFSITE

J. H. Carlson
U.S. Department of Energy
1000 Independence Avenue, SW
Washington, DC 20585

C. W. Conner
U.S. Department of Energy
1000 Independence Avenue, SW
Washington, DC 20585

K. A. Klein
U.S. Department of Energy
1000 Independence Avenue, SW
Washington, DC 20585

L. Marks
U.S. Department of Energy
1000 Independence Avenue, SW
Washington, DC 20585

M. L. Payton
U.S. Department of Energy
1000 Independence Avenue, SW
Washington, DC 20585

E. L. Wilmot
U.S. Department of Energy
1000 Independence Avenue, SW
Washington, DC 20585

Cindy Boggs-Mayes
Chicago Operations Office
U.S. Department of Energy
9800 S. Cass Avenue
Argonne, IL 60439

P. J. Gross
Oak Ridge Operations Office
U.S. Department of Energy
Oak Ridge, TN 37830

No. of Copies

M. A. Heiskell
Oak Ridge Operations Office
U.S. Department of Energy
Oak Ridge, TN 37830

L. Schappert
Oak Ridge National Laboratory
P.O. Box X
Oak Ridge, TN 37830

J. W. Cashwell
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185

R. Luna
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185

S. Gupta
Battelle-Columbus Laboratories
OSTP Department
505 King Ave
Columbus, OH 43201

R. Peterson
Battelle-Columbus Laboratories
OSTP Department
505 King Ave.
Columbus, OH 43201

30 DOE Technical Information Center

No. of Copies

ONSITE

5 DOE Richland Operations Office

M. Dayani
R. B. Goranson
R. D. Izatt
D. C. Langstaff
J. J. Sutey

Boeing Computer Services, Richland

M. E. Buxbaum

53 Pacific Northwest Laboratory

R. C. Adams
A. J. Boegel
J. L. Braitman
A. D. Chockie
L. L. Clark
C. A. Counts
J. W. Currie

No. of Copies

P. M. Daling
R. L. Engel
J. F. Fletcher
R. M. Gale
R. E. Heineman
G. M. Holter
C. H. Imhoff
P. N. McDuffie
R. W. McKee
G. W. McNair
D. F. Newman
D. R. Payson
K. J. Schneider
G. H. Sewart
M. R. Shay (20)
R. I. Smith
M. B. Triplett
R. C. Walling
M. K. White
T. W. Wood
Publishing Coordination MH (2)
Technical Information (5)