SY28-0766-0
File No. S370-36

Systems

OS/VS2
System Logic Library
Volume 6

VS2.03.804
VS2.03.805
VS2.03.807

IBM

Pages numbered as duplicates in this publication must be retained because each of these pages contains information specific to an individual Selectable Unit.

This minor revision incorporates the following Selectable Units:

Scheduler Improvements	VS2.03.804
Supervisor Performance #1	VS2.03.805
Supervisor Performance #2	VS2.03.807

The selectable unit to which the information applies, is noted in the upper corner of the page.

First Edition (July, 1976)

This is a reprint of SY28-0718-0 incorporating changes released in the following Selectable Unit Newsletters:

SN28-2684 (dated May 28, 1976)
SN28-2689 (dated May 28, 1976)
SN28-2695 (dated May 28, 1976)

This edition applies to Release 3.7 of OS/VS2 and to all subsequent releases of OS/VS2 until otherwise indicated in new editions or Technical Newsletters. Changes are continually made to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370 Bibliography,* GC20-0001, for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Publications Development, Department D58, Building 706-2, PO Box 390, Poughkeepsie, N.Y. 12602. Comments become the property of IBM.

System Logic Library comprises seven volumes. Following is the content and order number for each volume.

OS/VS2 System Logic Library,

Volume 1 contents: SY28-0713
- MVS logic introduction
- Abbreviation list
- Index for all volumes

Volume 2 contents: SY28-0714
- Method of Operation diagrams for
- Communications Task
- Command Processing
- Region Control Task (RCT)
- Started Task Control (STC)
- LOGON Scheduling

Volume 3 contents: SY28-0715
- Method of Operation diagrams for
- System Resources Manager (SRM)
- System Activity Measurement Activity (MF/1)
- JOB Scheduling
  - —Subsystem Interface
  - —Master Subsystem
  - —Initiator/Terminator
  - —SWA Create Interface
  - —Converter/Interpreter
  - —SWA Manager
  - —Allocation/Unallocation
  - —System Management Facilities (SMF)
  - —System Log
  - —Checkpoint/Restart

Volume 4 contents: SY28-0716
- Method of Operation diagrams for
- Timer Supervision
- Supervisor Control
- Task Management
- Program Management
- Recovery/Termination Management (R/TM)

Volume 5 contents: SY28-0717
- Method of Operation diagrams for
- Real Storage Management (RSM)
- Virtual Storage Management (VSM)
- Auxiliary Storage Management (ASM)

Volume 6 contents: SY28-0718
- Program Organization

Volume 7 contents: SY28-0719
- Directory
- Data Areas
- Diagnostic Aids

Please note that if you use only one order number, you will only receive that volume. To receive all seven volumes, you must either use all seven form numbers or, simply the following number: SBOF-8210. If you use SBOF-8210, you will receive all seven volumes.

The publication is intended for persons who are debugging or modifying the system. For general information about the use of the MVS system, refer to the publication *Introduction to OS/VS Release 2,* GC28-0661.

How This Publication is Organized

This publication contains six chapters. Following, is a synopsis of the information in each section:

- *Introduction and Master Index* — an overview of each of the functions this publication documents, an abbreviation list of all acronyms used in the publication, and a complete index for all seven volumes.
- *Method of Operation* — a functional approach to each of the subcomponents, using both diagrams and text. Each subcomponent begins with an introduction; all the diagrams and text applying to that subcomponent follow.
- *Program Organization* — a description of module-to-module flow for each subcomponent; a description of each module's function, including entry and exit. The module-to-module flow is ordered by subcomponent. The module descriptions are in alphabetic order without regard to subcomponent.
- *Directory* — a cross-reference from names in the various subcomponents to their place in the source code and in the publication.
- *Data Areas* — a description of the major data areas used by the subcomponents (only those, however, that are not described in *OS/VS Data Areas,* SYB8-0606, which is on microfiche); a data area usage table, showing whether a module reads or updates a data area; a control block overview diagram for each subcomponent, showing the various pointer schemes for the control blocks applicable to each subcomponent; a table detailing data area acronyms, mapping macro instructions, common names, and symbol usage table.

- *Diagnostic Aids* — the messages issued, including the modules that issue, detect, and contain the message; register usage; return codes; wait state codes; and miscellaneous aids.

Corequisite Reading

The following publications are corequisites:

- *OS/VS2 JES2 Logic,* SY28-0622
- *OS/VS Data Areas,* SYB8-0606 (This document is on microfiche.)
- *OS/VS2 System Initialization Logic,* SY28-0623

Contents

Figures

Section 3: Program Organization

This section describes the physical organization of the object modules that perform the scheduler and supervisor functions for the OS/VS2 operating system. The first part of the section is a collection of charts showing the control flow among the object modules during the operation of the OS/VS2 system. The charts are arranged by subcomponent. The second part describes each object module as follows:

- Summarizes the operation of the module.
- Names the modules that pass control to it.
- Names the modules that receive control from it.

The module descriptions are arranged alphamerically by module name.

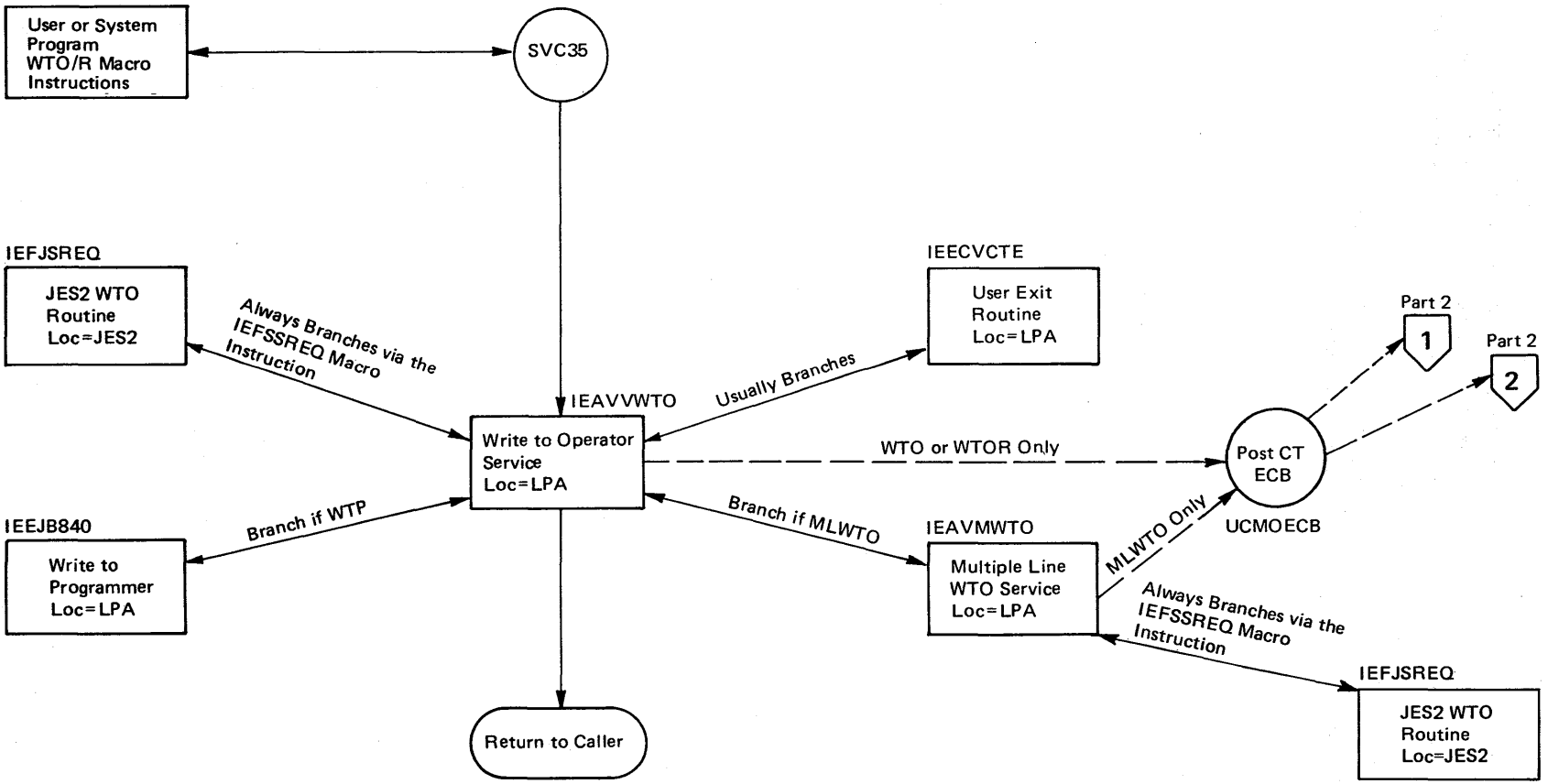


Figure 3-1. WTO and WTOR Processing Module Flow (Communication Task) (Part 1 of 2)

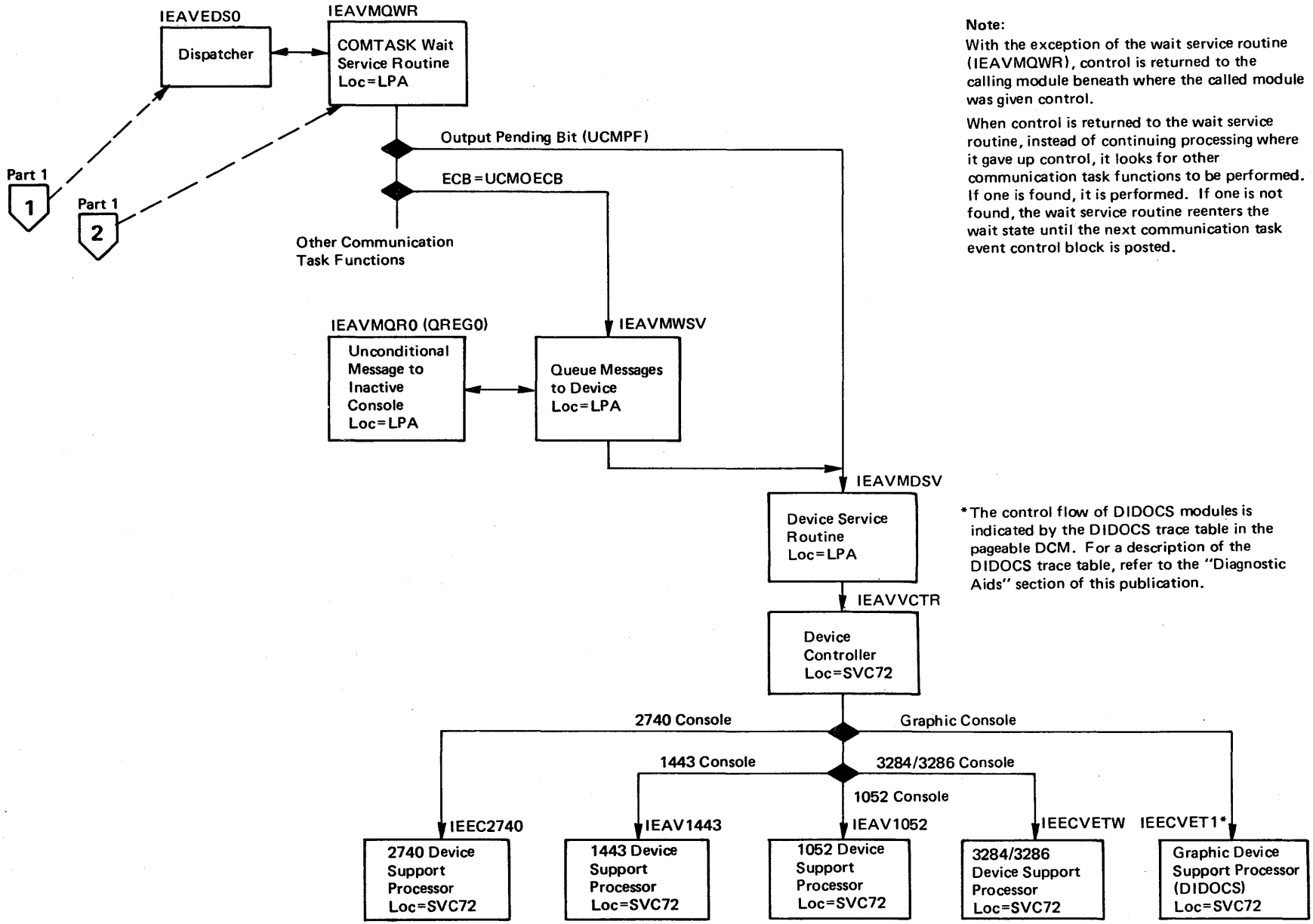
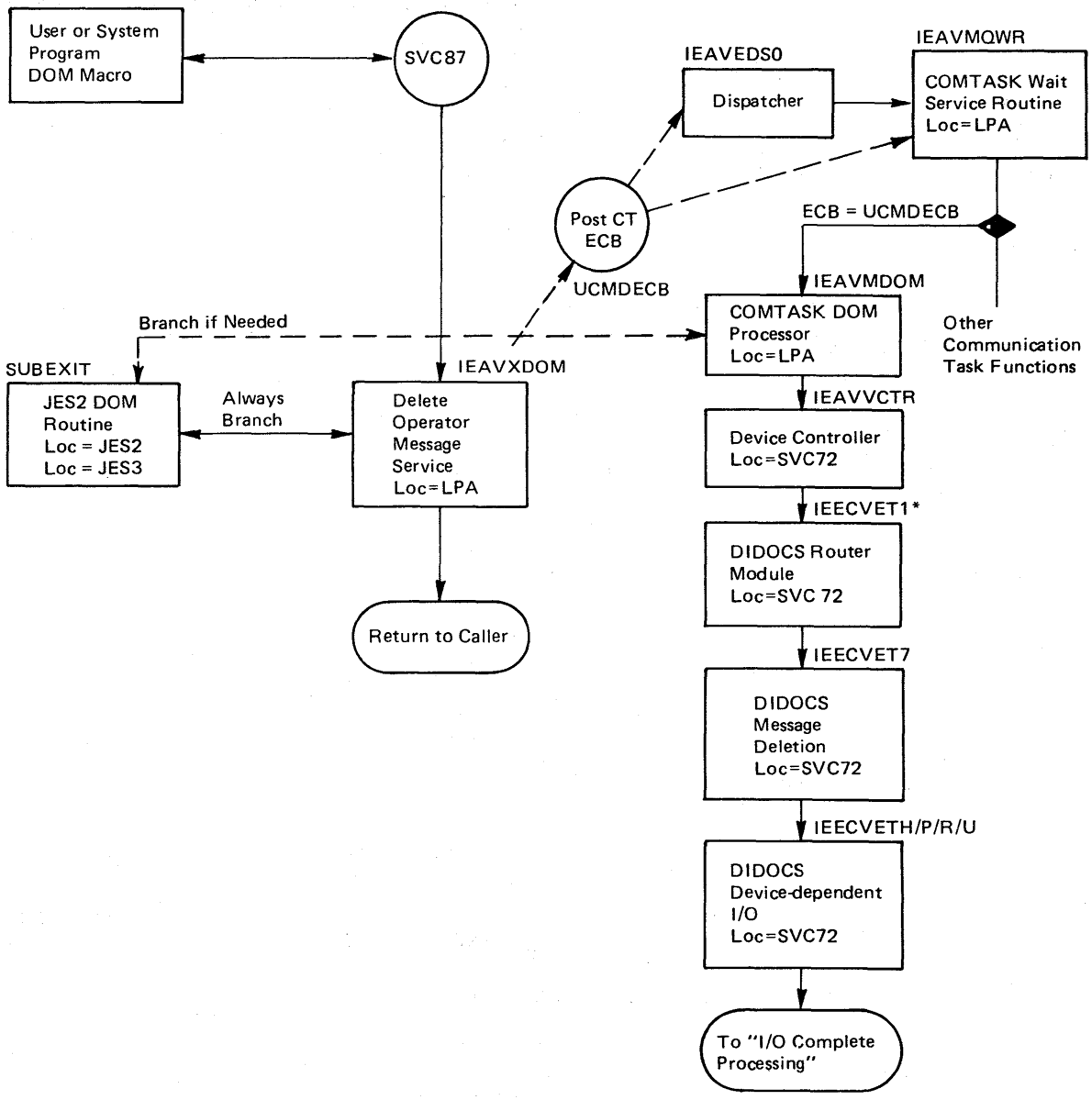


Figure 3-1. WTO and WTOR Processing Module Flow (Communication Task) (Part 2 of 2)



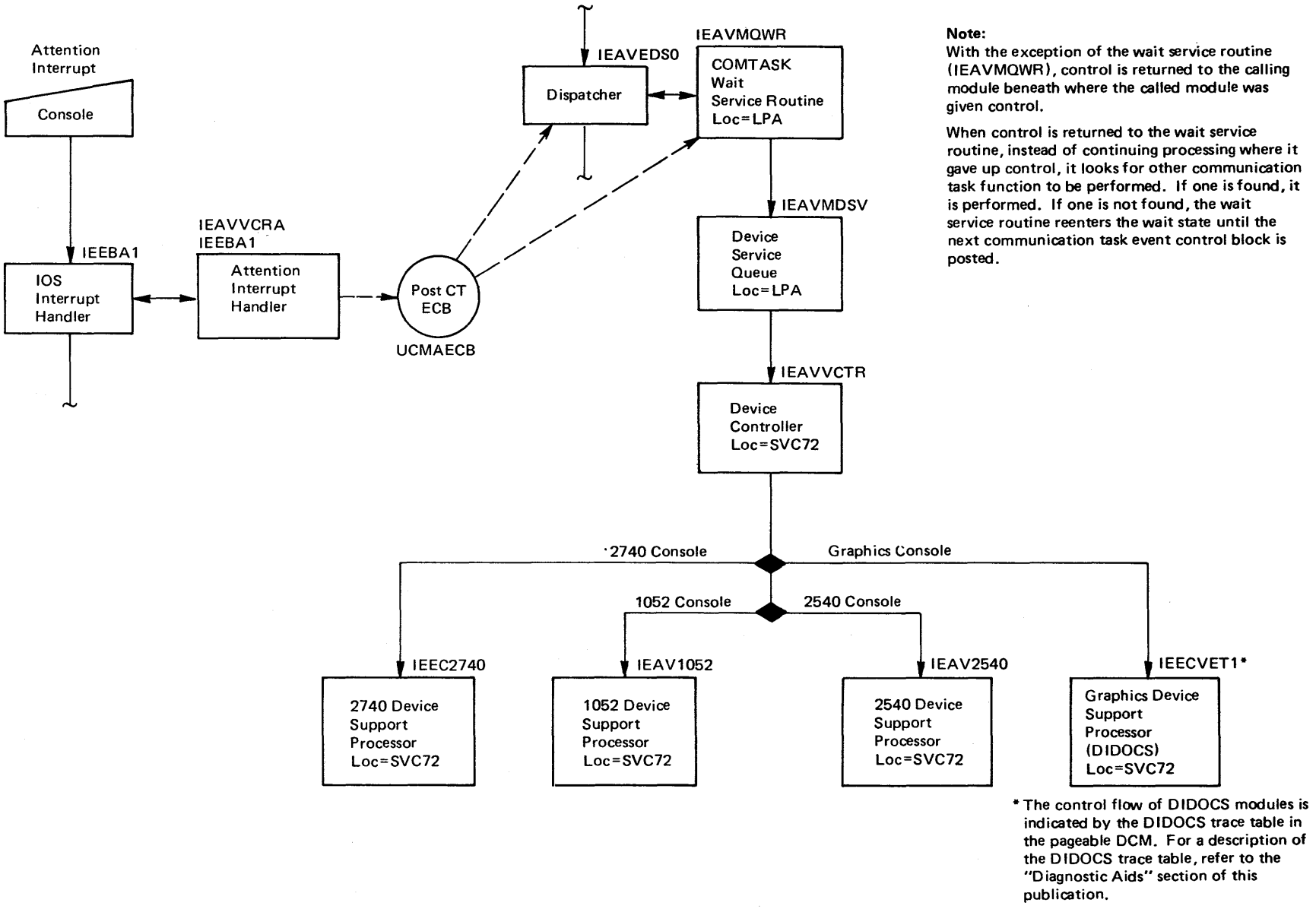
Note:

With the exception of the wait service routine (IEAVMQWR), control is returned to the calling module beneath where the called module was given control.

When control is returned to the wait service routine, instead of continuing processing where it gave up control, it looks for other communication task functions to be performed. If one is found, it is performed. If one is not found, the wait service routine reenters the wait state until the next communication task event control block is posted.

*The control flow of DIDOCS modules is indicated by the DIDOCS trace table in the pageable DCM. For a description of the DIDOCS trace table, refer to the "Diagnostic Aids" section of this publication.

Figure 3-2. DOM Processing Module Flow (Communication Task)

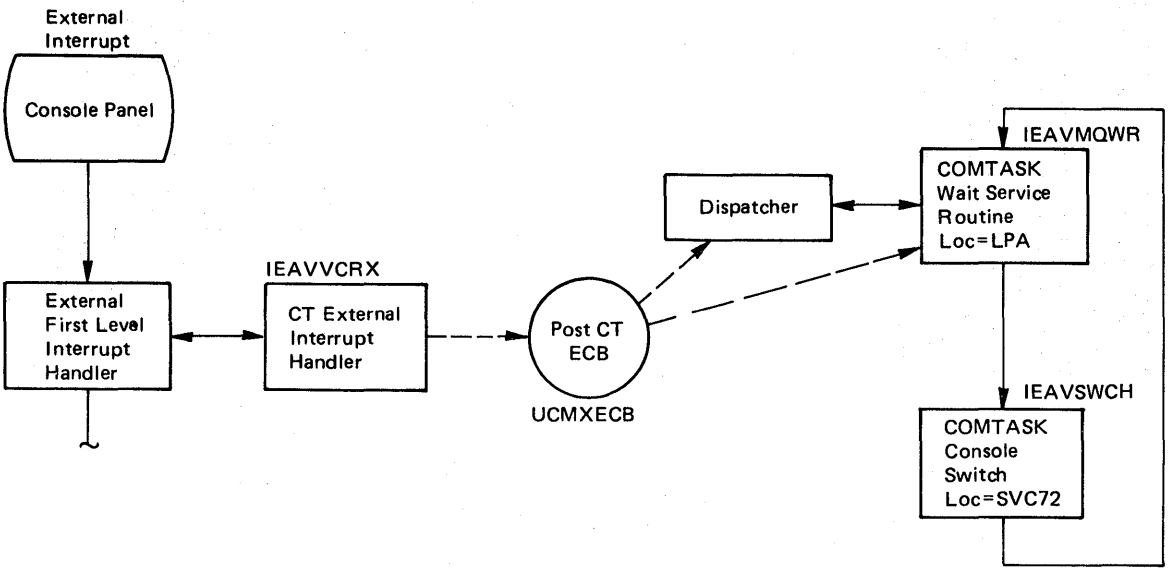


Note:
With the exception of the wait service routine (IEAVMQWR), control is returned to the calling module beneath where the called module was given control.

When control is returned to the wait service routine, instead of continuing processing where it gave up control, it looks for other communication task function to be performed. If one is found, it is performed. If one is not found, the wait service routine reenters the wait state until the next communication task event control block is posted.

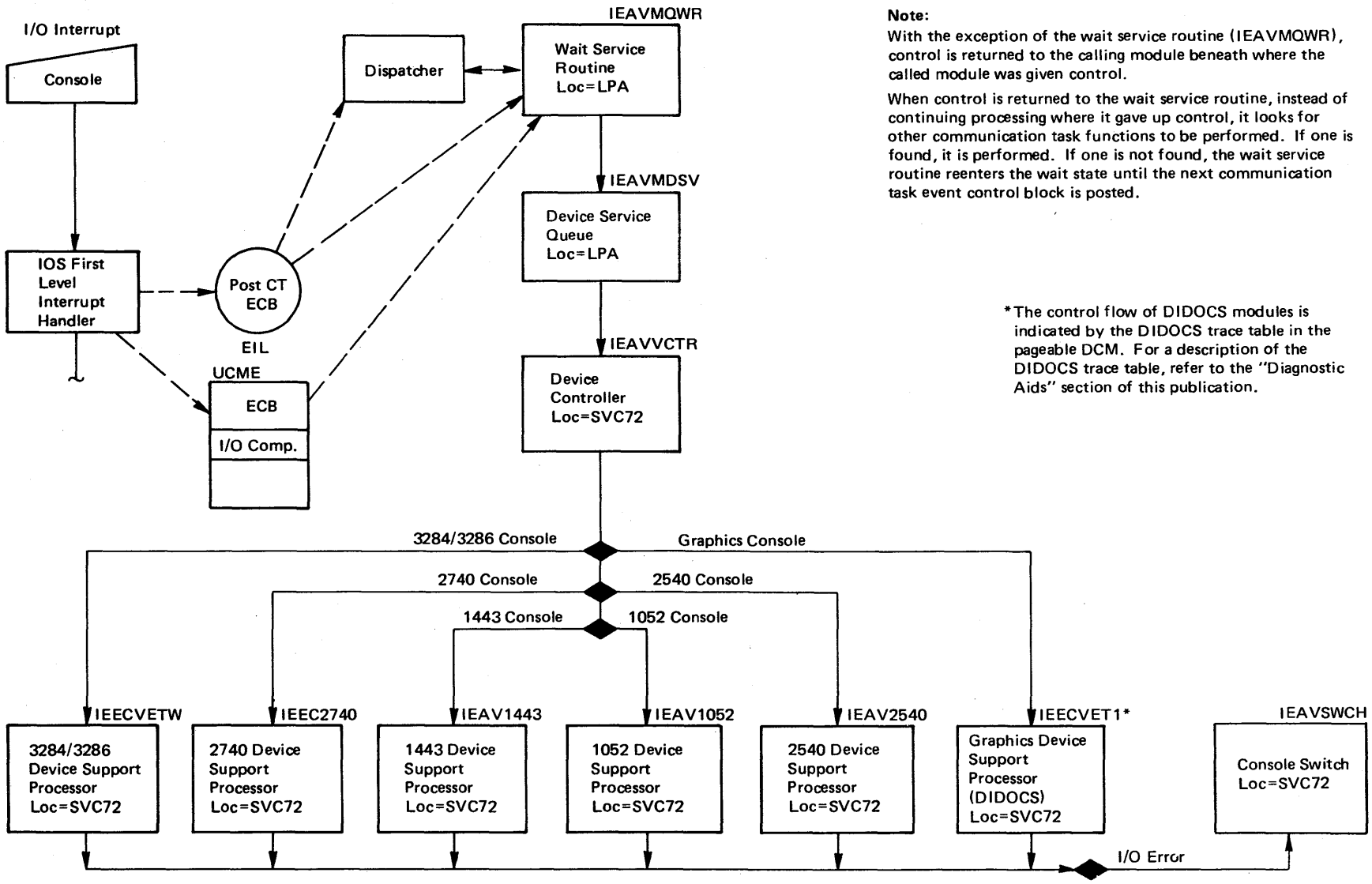
* The control flow of DIDOCS modules is indicated by the DIDOCS trace table in the pageable DCM. For a description of the DIDOCS trace table, refer to the "Diagnostic Aids" section of this publication.

Figure 3-3. Attention Interrupt Processing Module Flow (Communication Task)



Note: On return to the Wait Service Routine, if there are no other services needed, the wait service routine reenters the wait state until the next communication task event control block is posted.

Figure 3-4. External Interrupt Processing Module Flow (Communication Task)



Note:

With the exception of the wait service routine (IEAVMQWR), control is returned to the calling module beneath where the called module was given control.

When control is returned to the wait service routine, instead of continuing processing where it gave up control, it looks for other communication task functions to be performed. If one is found, it is performed. If one is not found, the wait service routine reenters the wait state until the next communication task event control block is posted.

*The control flow of DIDOCS modules is indicated by the DIDOCS trace table in the pageable DCM. For a description of the DIDOCS trace table, refer to the "Diagnostic Aids" section of this publication.

Figure 3-5. I/O Complete Processing Module Flow (Communication Task)

Legend:

A	ATTACH
BR	Branch
C	CALL
D	Schedule an SRB routine.
(E)	Diagnostic return to IEE0503D, then to return point (R)
(EOT)	End-of-task processing (Returns to system, for attached tasks).
(F)	Go to IEECB860 to set up recovery for attached processors, via LOAD/BALR/DELETE.
(G)	Diagnostic return for graphics-oriented commands (those with L=cca operand) to the following: IEE5603D → IEE5903D → (R); IEE5603D → (R); IEE5603D → IEE0503D → (R) (If storage is unavailable)
L	LOAD
M	Most errors.
P	POST macro instruction.
(R)	Return to IEE0003D, then to Supervisor SVC Handler. **Note:** The (R) shown by IEE0803D applies after the posting of module IEEVWAIT.
S	Some errors.
WAIT	Master Scheduler wait routine, IEEVWAIT.
X	XCTL
(xxx) (xxx,yyy)	Command operand(s).
1	The NET operand currently is unsupported.
2	See TCAM PLM for further logic details.
3	See RMS PLM for further logic details.

Notes:

- Module IEE0803D creates CSCB for commands that are to be attached by module IEEVWAIT as separate tasks.
- Module IEAVEMRQ initiates creation of a new memory (address space).
- Processors receiving control from module IEEVWAIT operate in the master scheduler's memory.
- After cross-memory posting the wait routine, module IEE0803D returns control to (R).
- All entries to, and exits from, module IEE0803D are via BR.

Figure 3-6. Command Processing Program Organization Overview (Part 1 of 14)

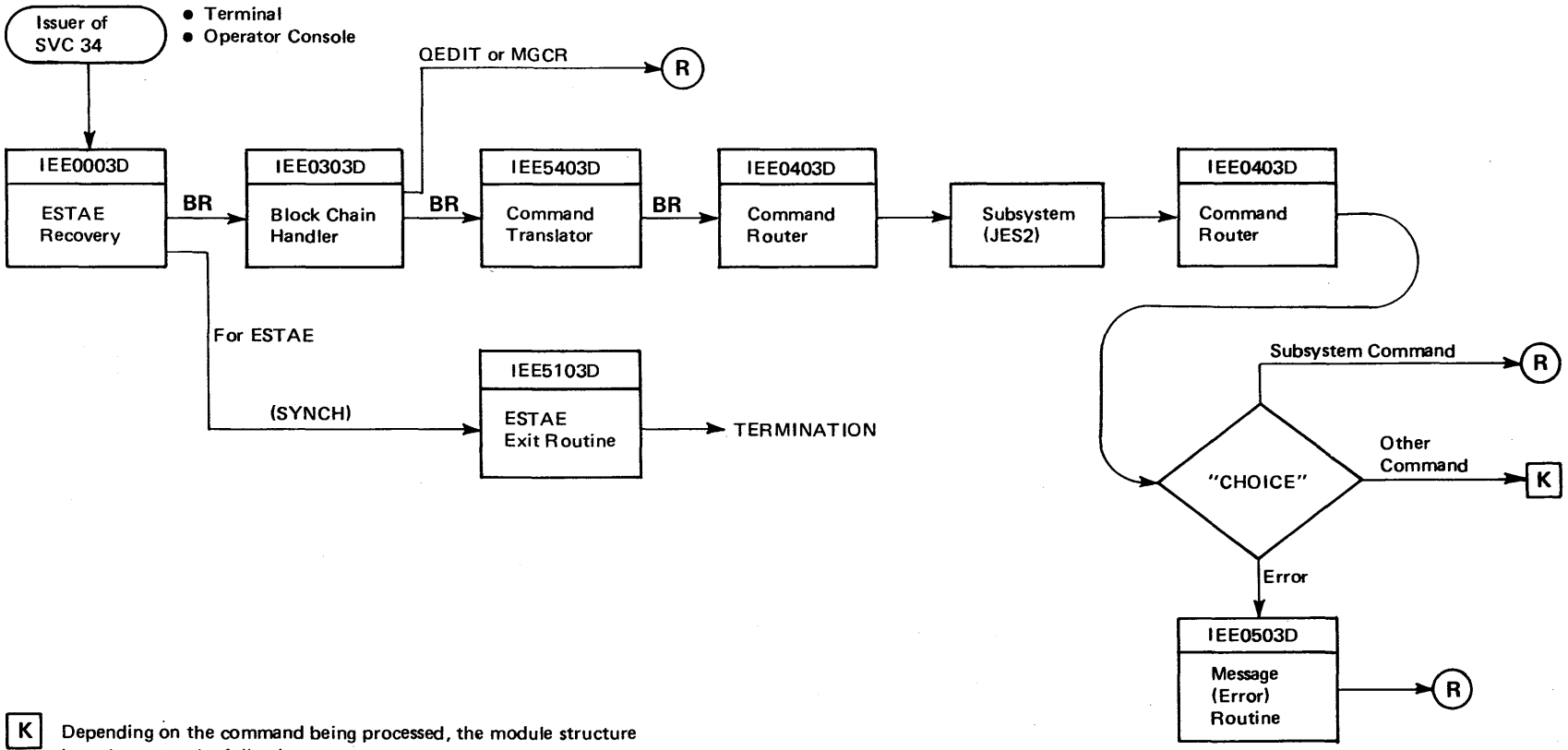


Figure 3-6. Command Processing Program Organization Overview (Part 2 of 14)

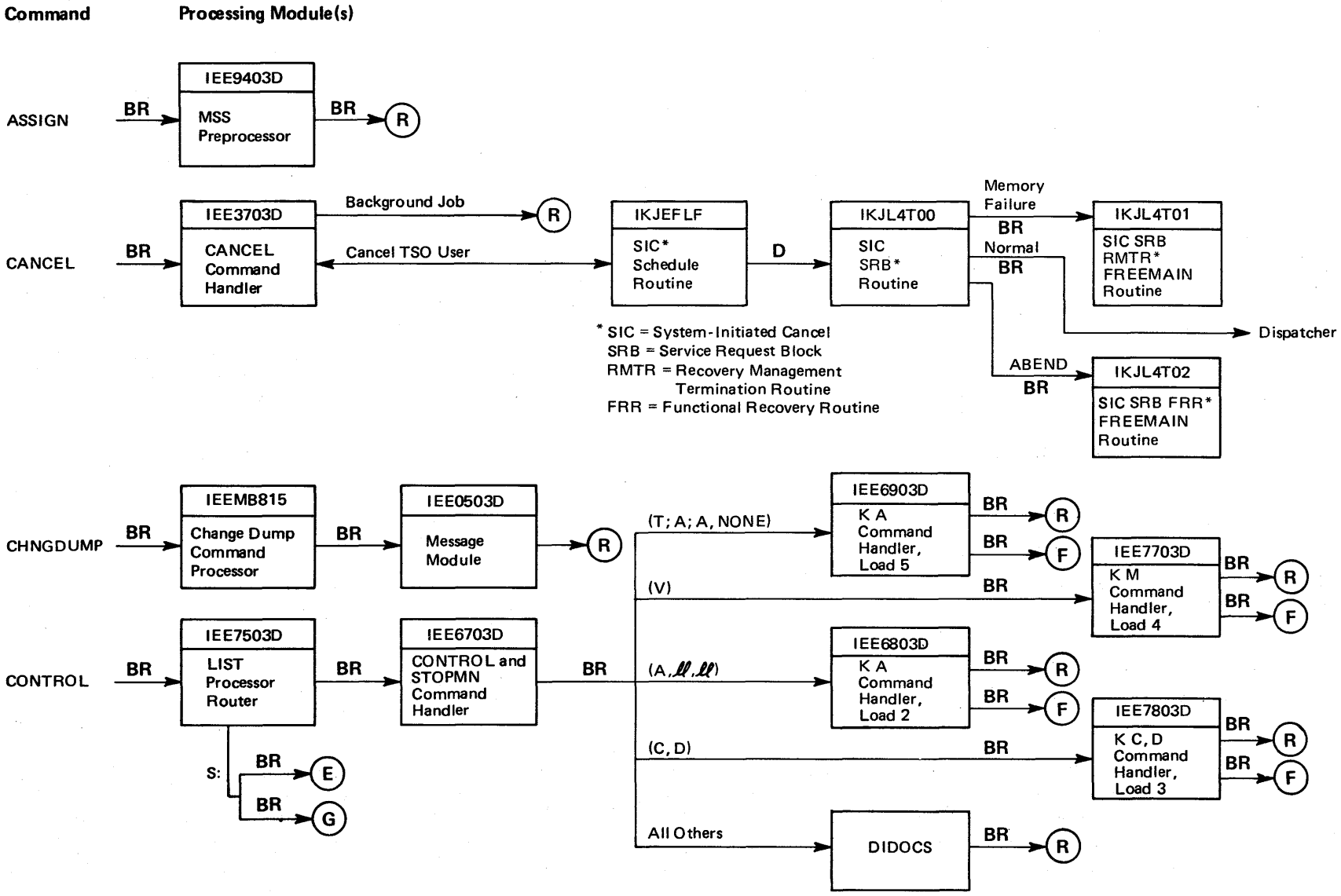


Figure 3-6. Command Processing Program Organization Overview (Part 3 of 14)

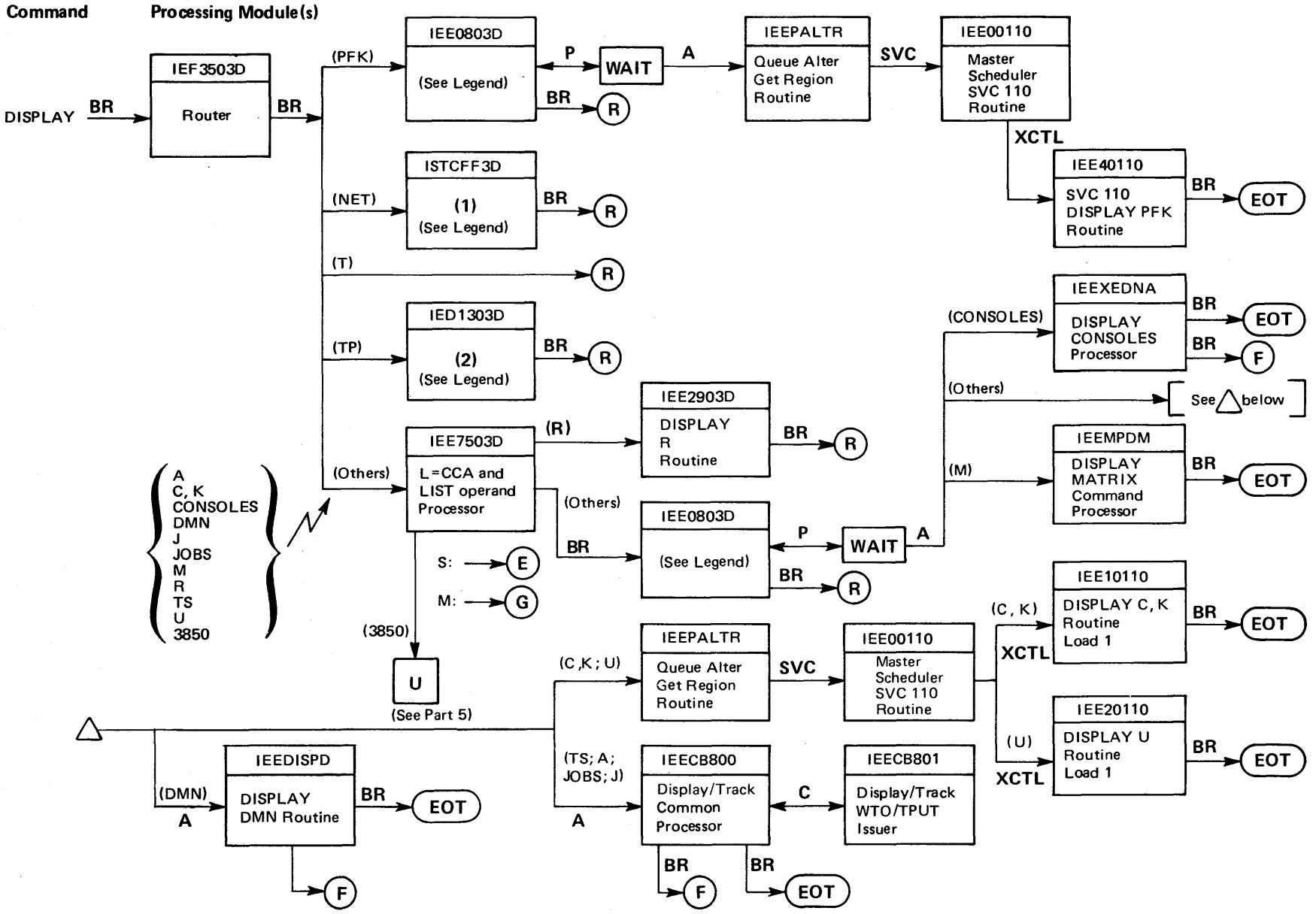


Figure 3-6. Command Processing Program Organization Overview (Part 4 of 14)

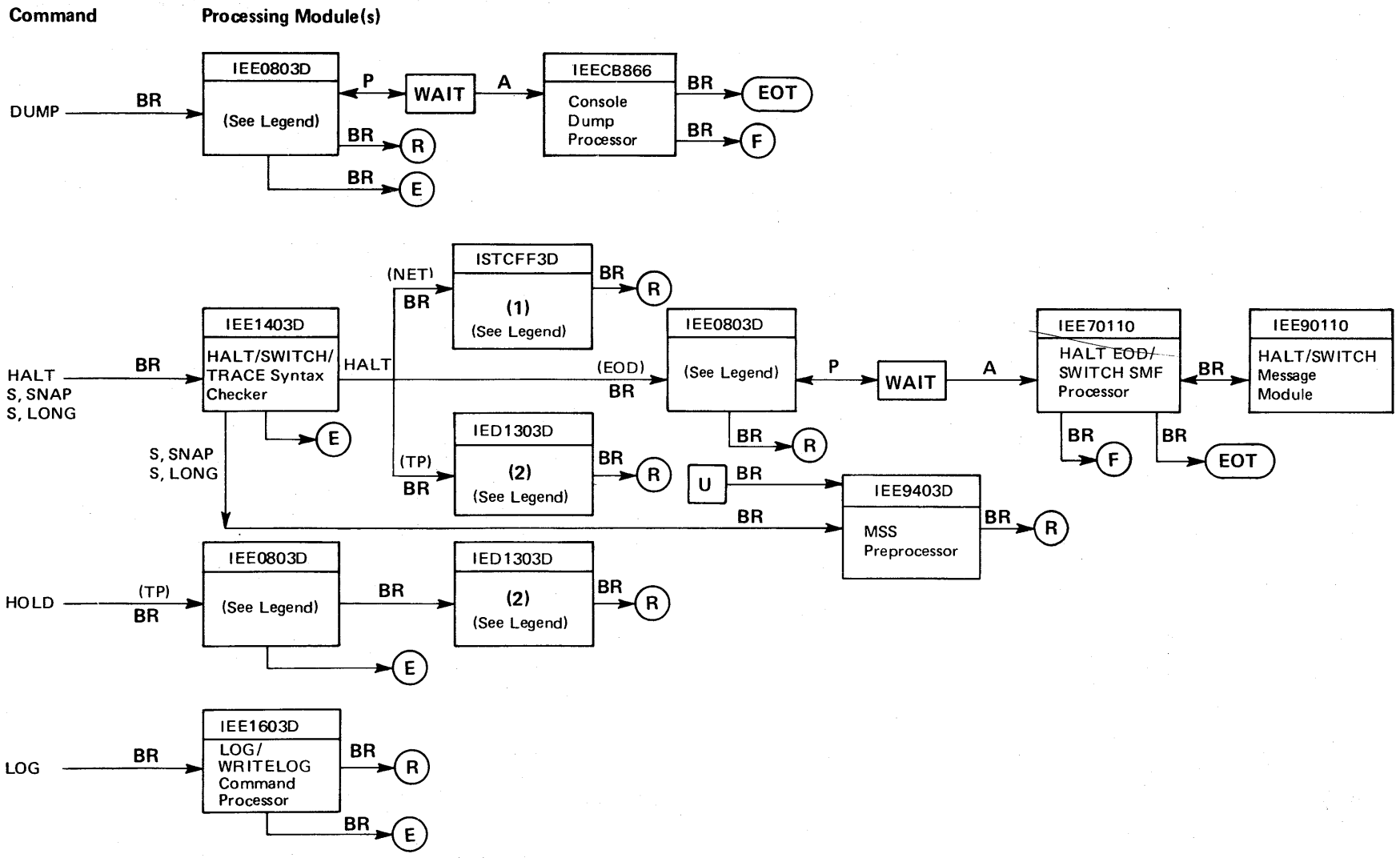


Figure 3-6. Command Processing Program Organization Overview (Part 5 of 14)

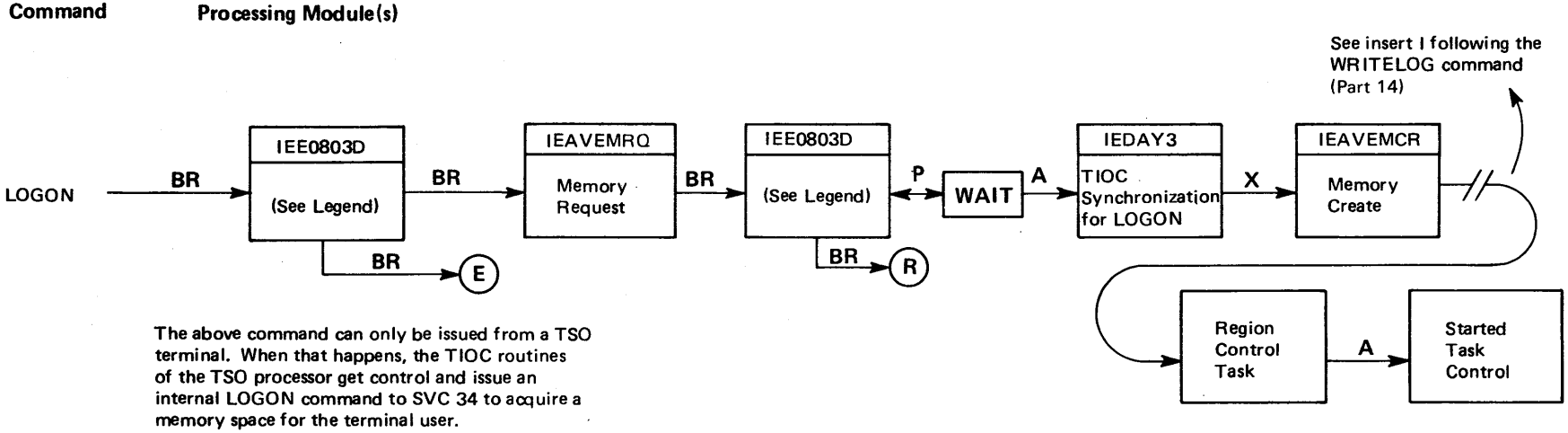


Figure 3-6. Command Processing Program Organization Overview (Part 6 of 14)

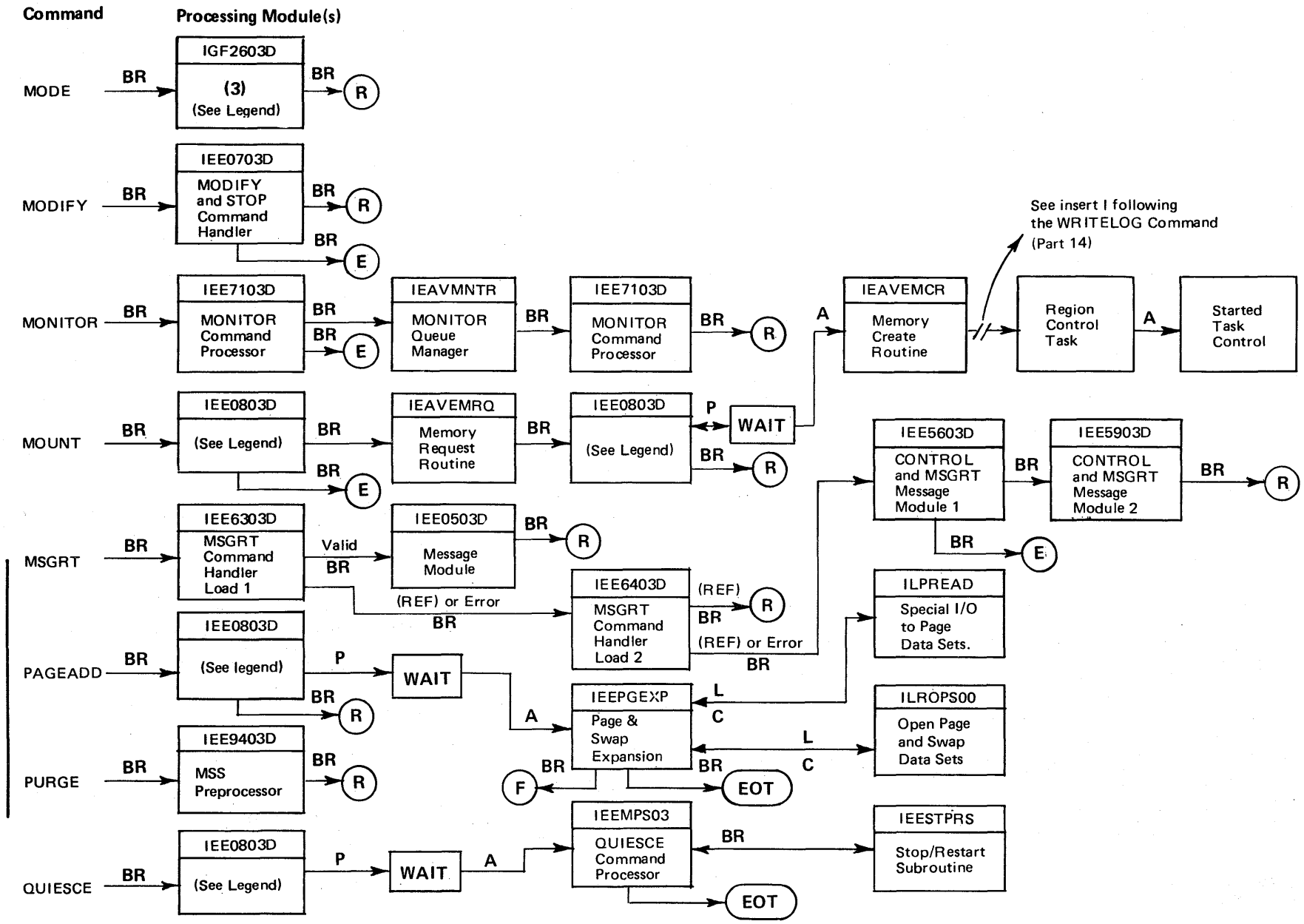


Figure 3-6. Command Processing Program Organization Overview (Part 7 of 14)

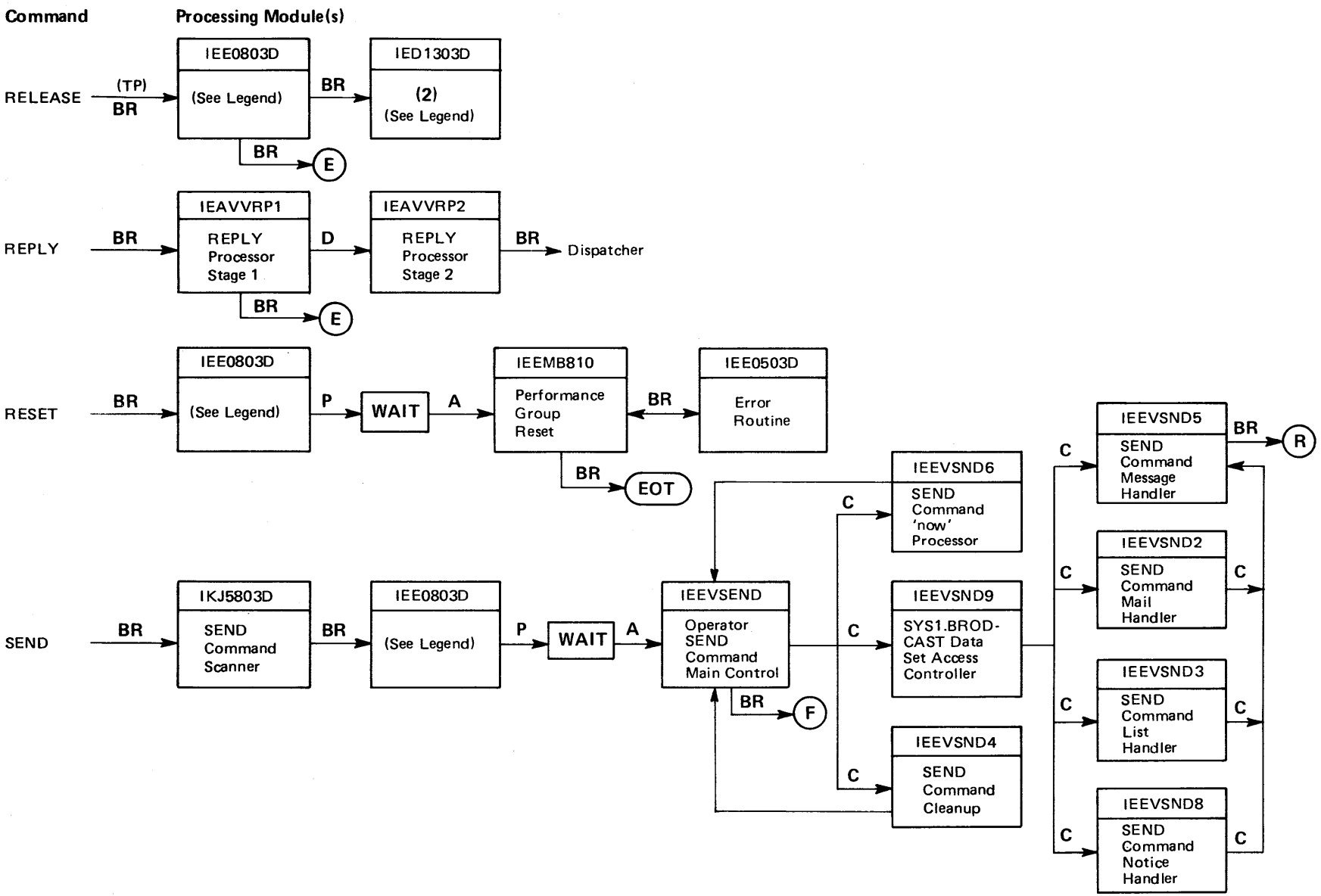


Figure 3-6. Command Processing Program Organization Overview (Part 8 of 14)

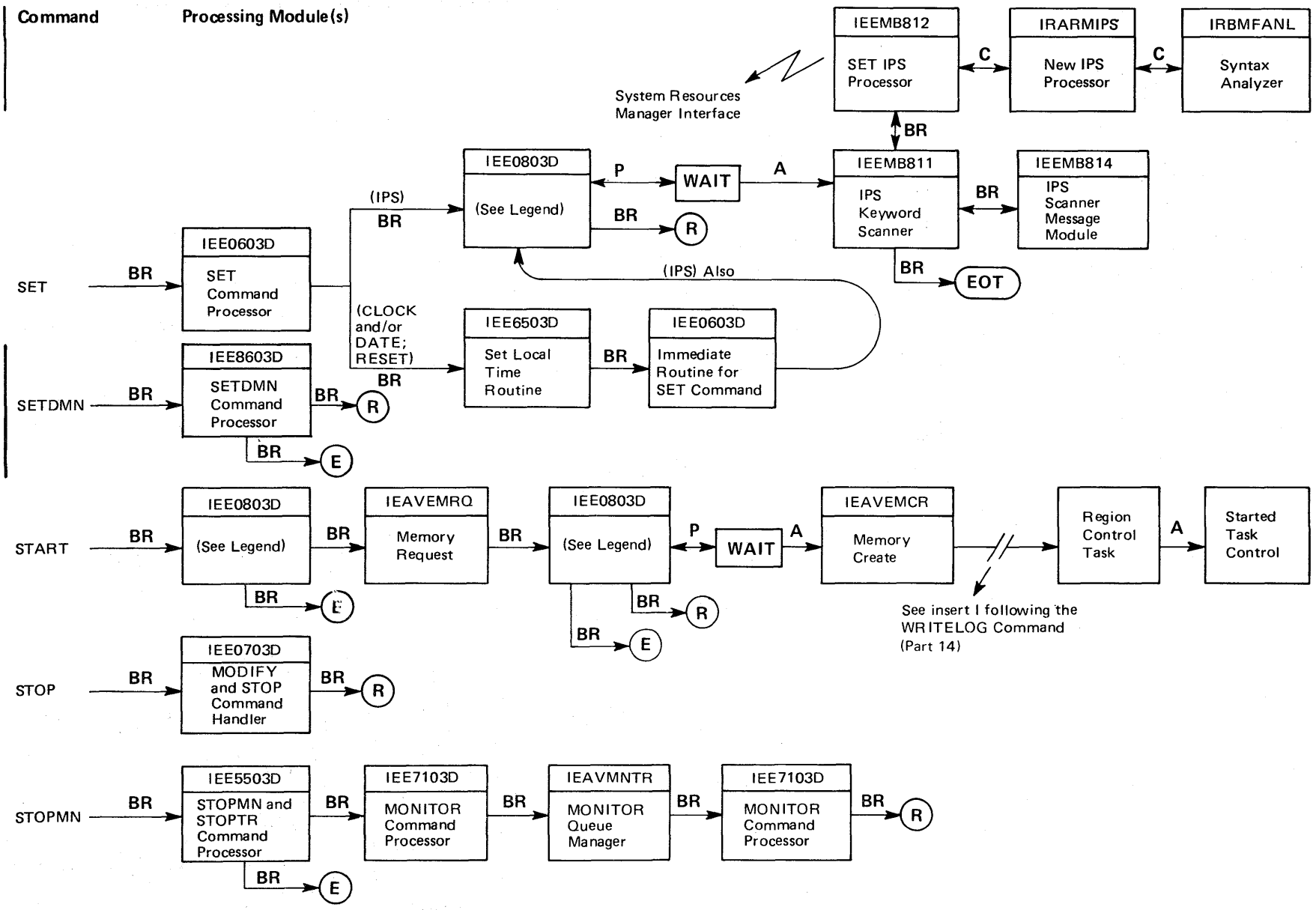


Figure 3-6. Command Processing Program Organization Overview (Part 9 of 14)

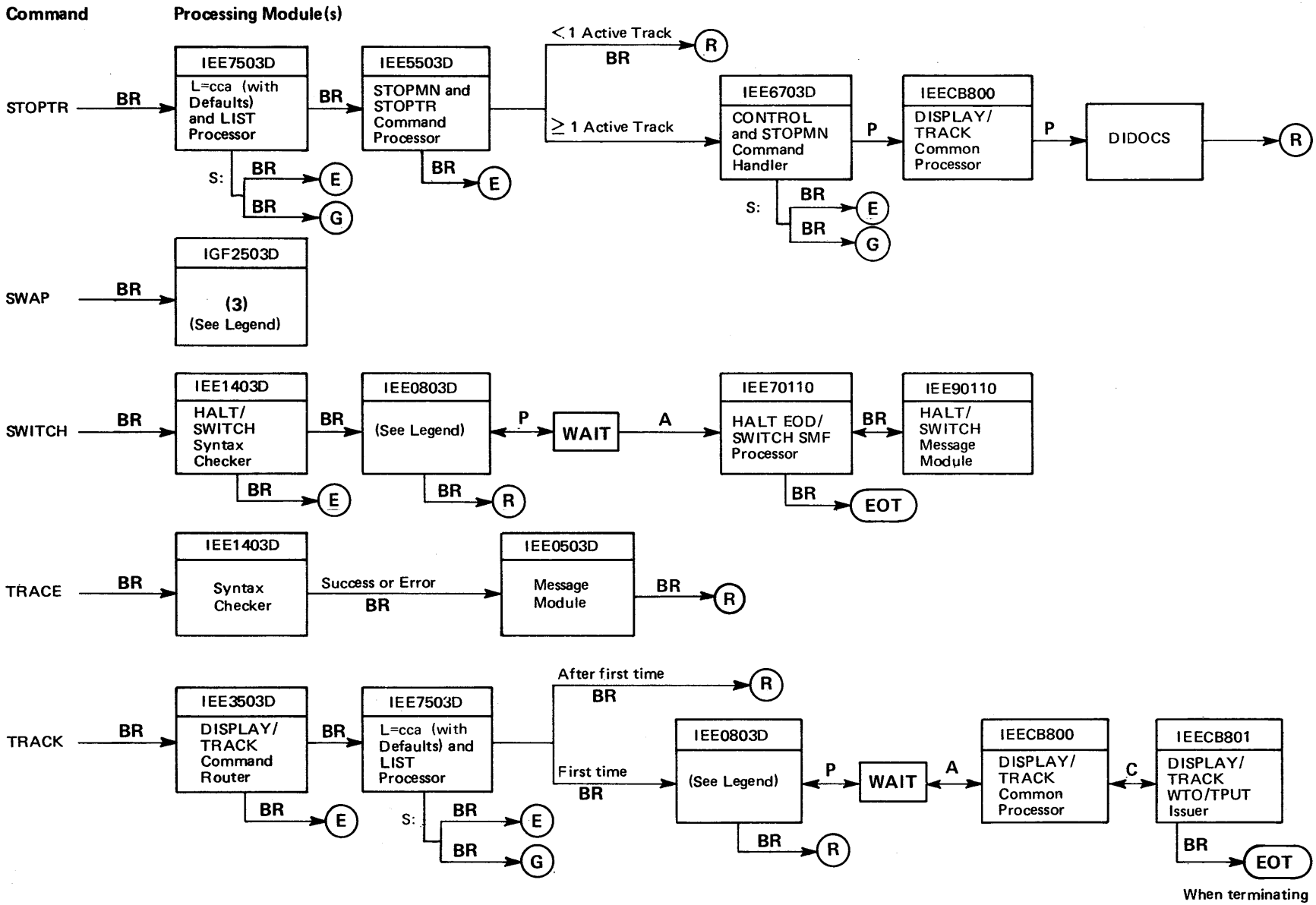


Figure 3-6. Command Processing Program Organization Overview (Part 10 of 14)

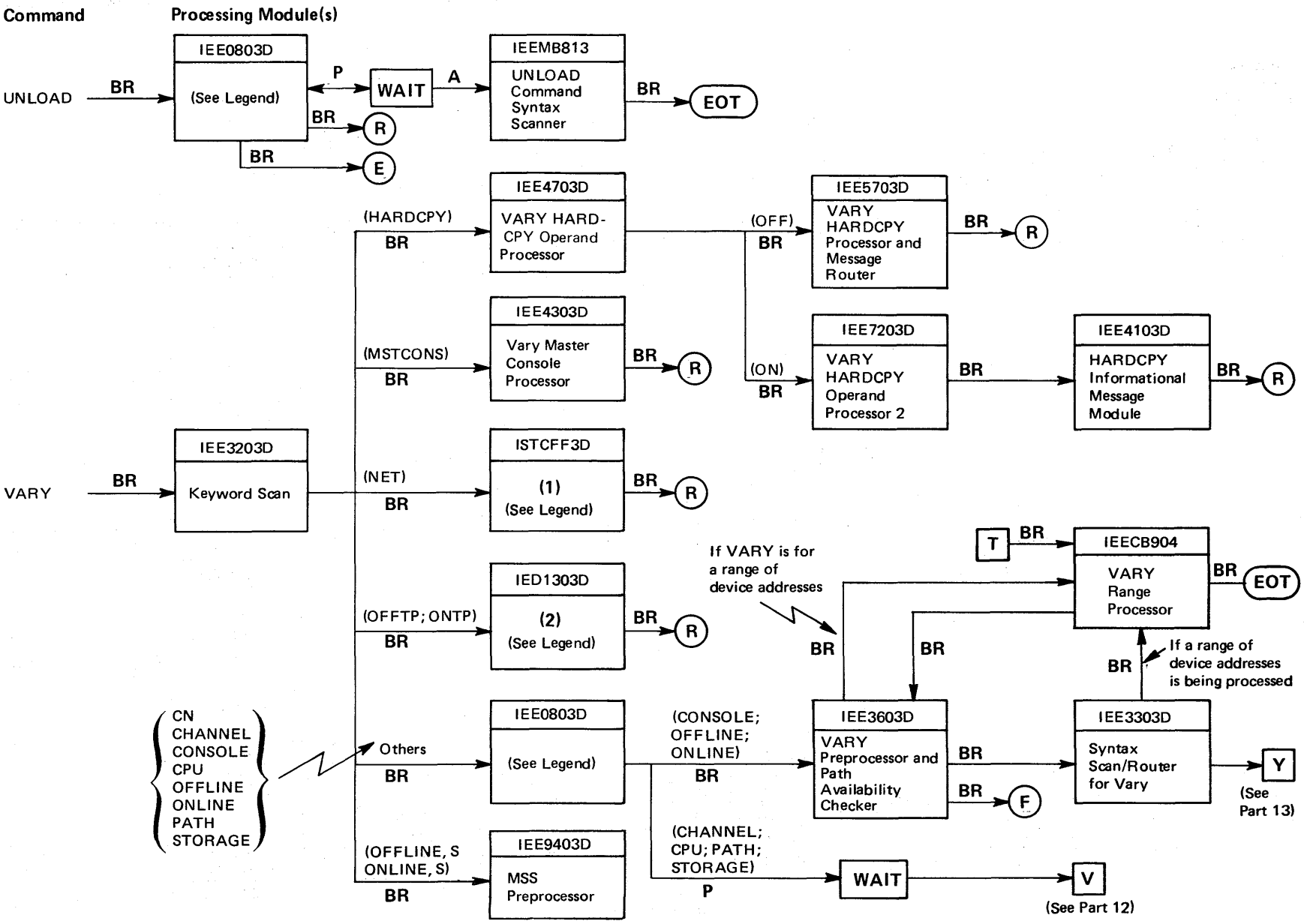


Figure 3-6. Command Processing Program Organization Overview (Part 11 of 14)

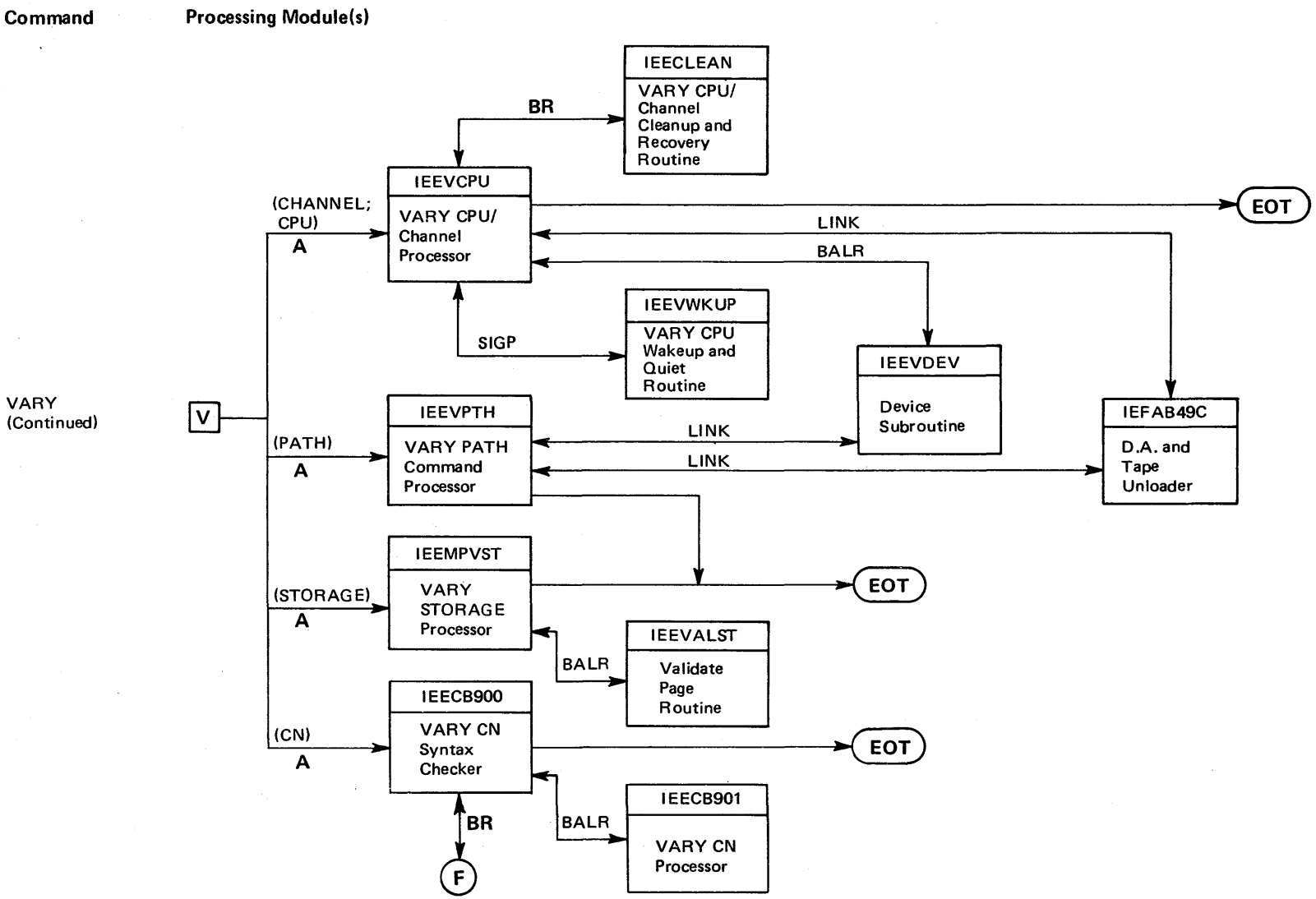


Figure 3-6. Command Processing Program Organization Overview (Part 12 of 14)

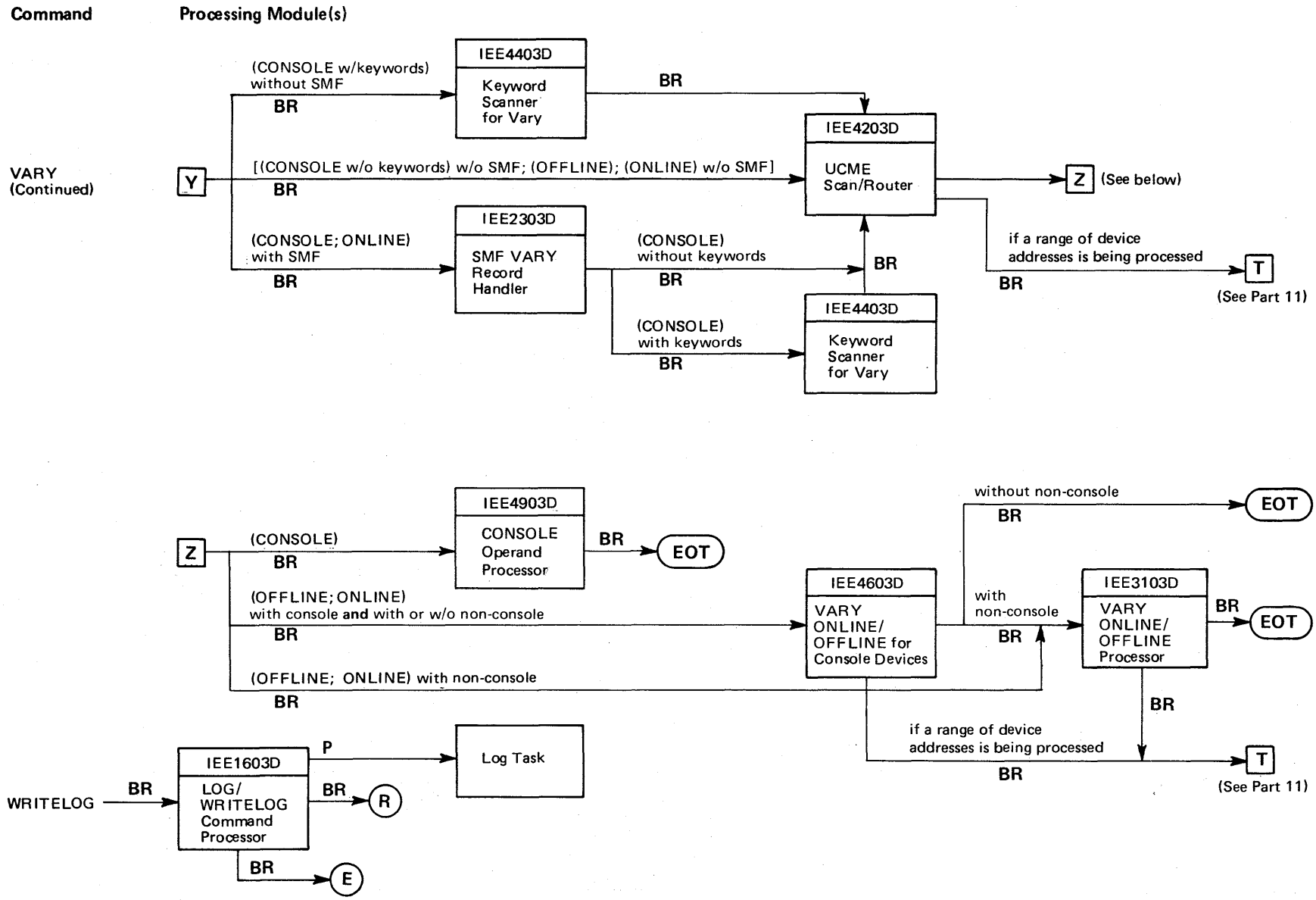


Figure 3-6. Command Processing Program Organization Overview (Part 13 of 14)

Insert I. This insert provides greater detail of the module flow for the address space creation process.

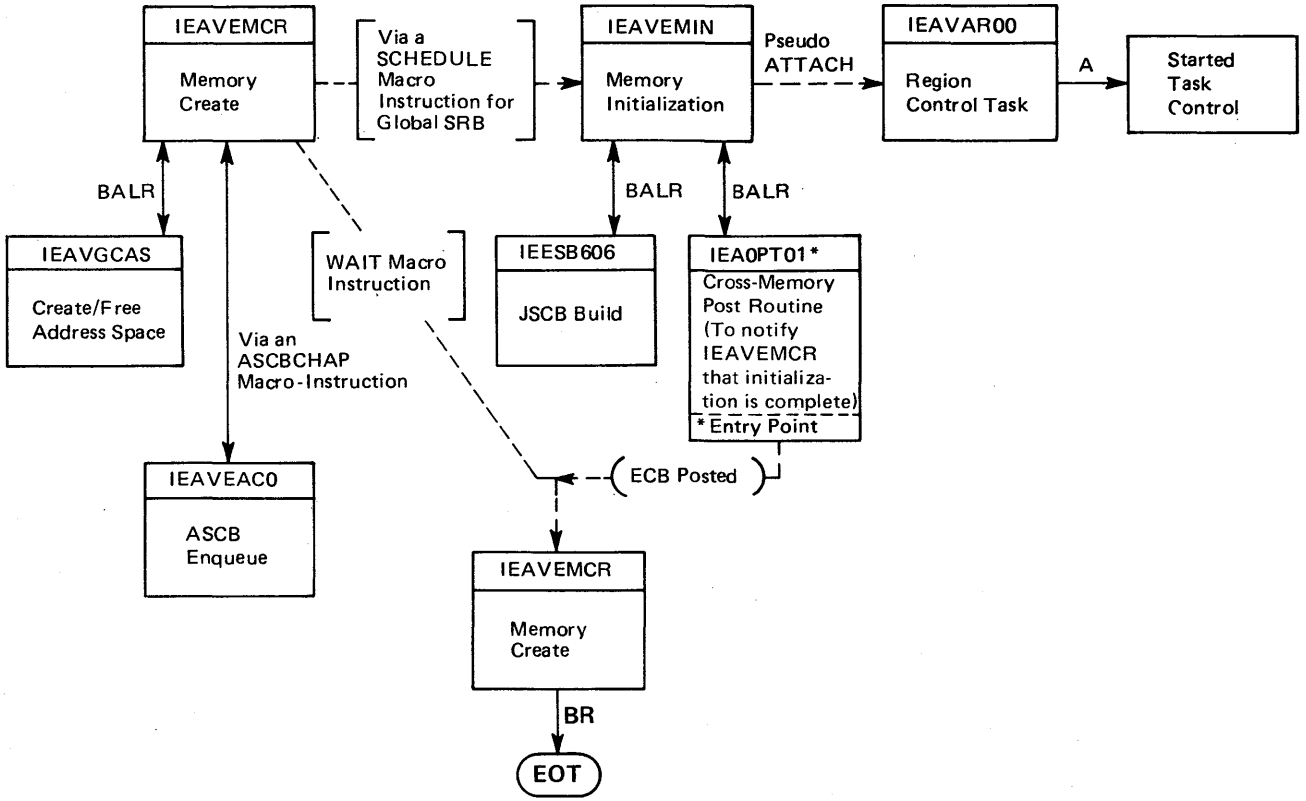


Figure 3-6. Command Processing Program Organization Overview (Part 14 of 14)

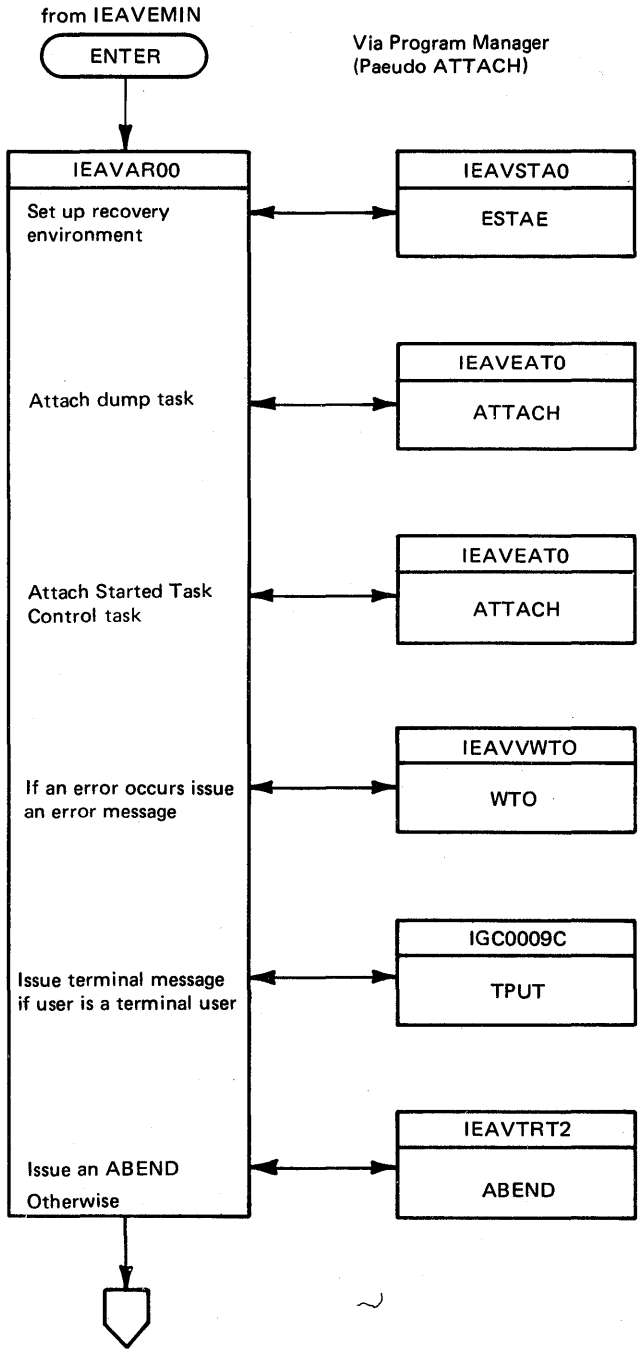


Figure 3-7. Region Control Task (RCT) Module Flow (Part 1 of 10)

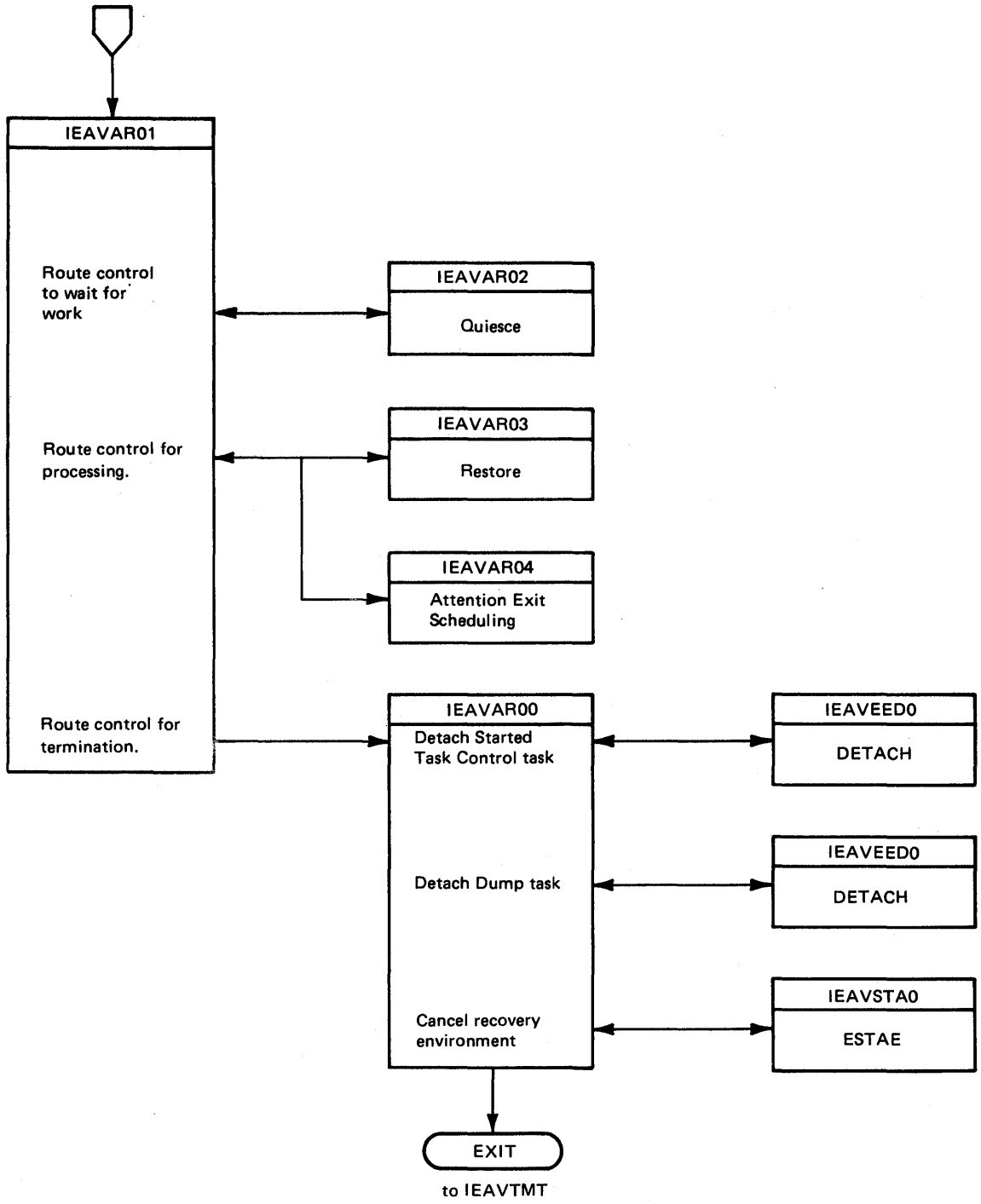


Figure 3-7. Region Control Task (RCT) Module Flow (Part 2 of 10)

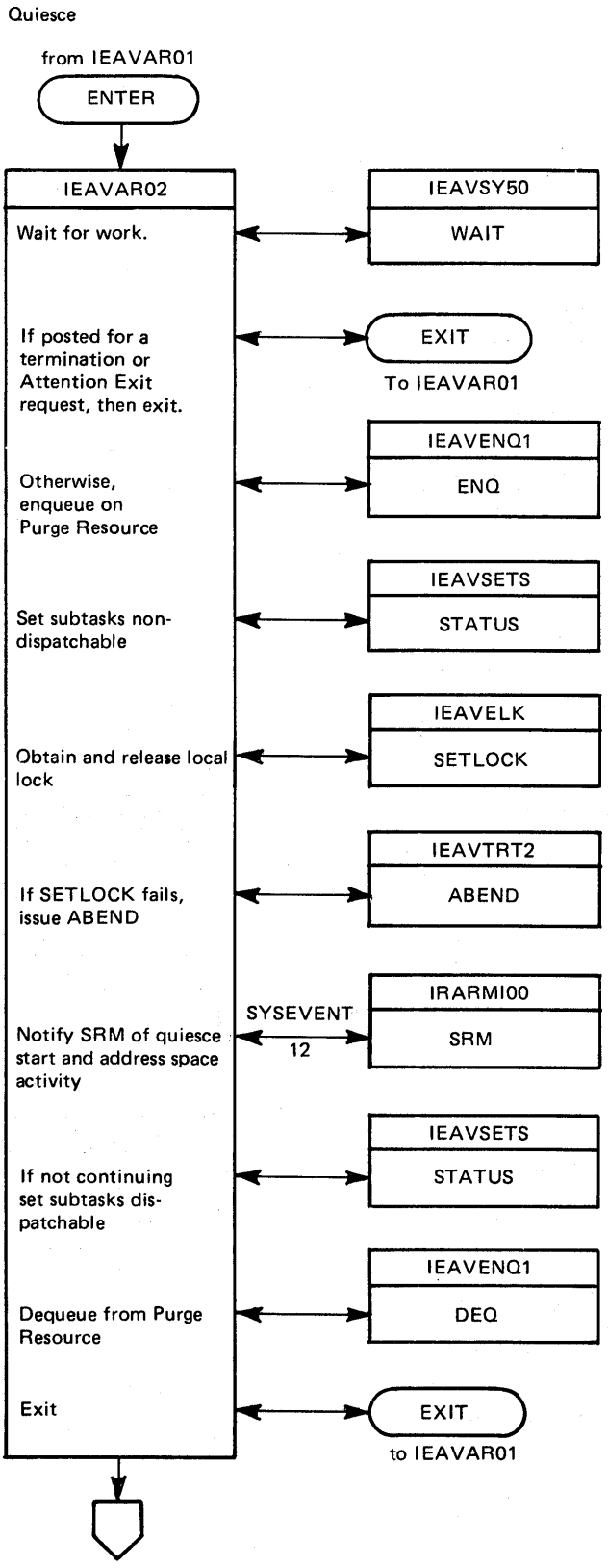


Figure 3-7. Region Control Task (RCT) Module Flow (Part 3 of 10)

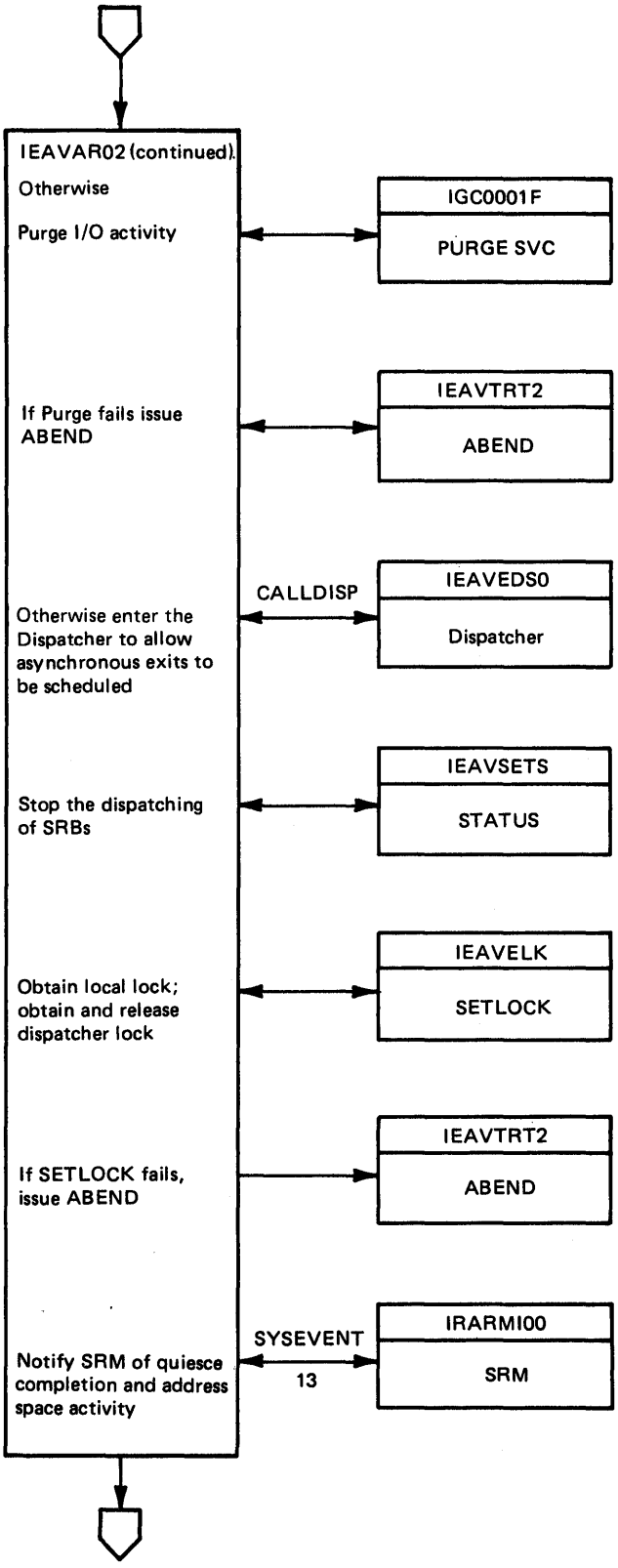


Figure 3-7. Region Control Task (RCT) Module Flow (Part 4 of 10)

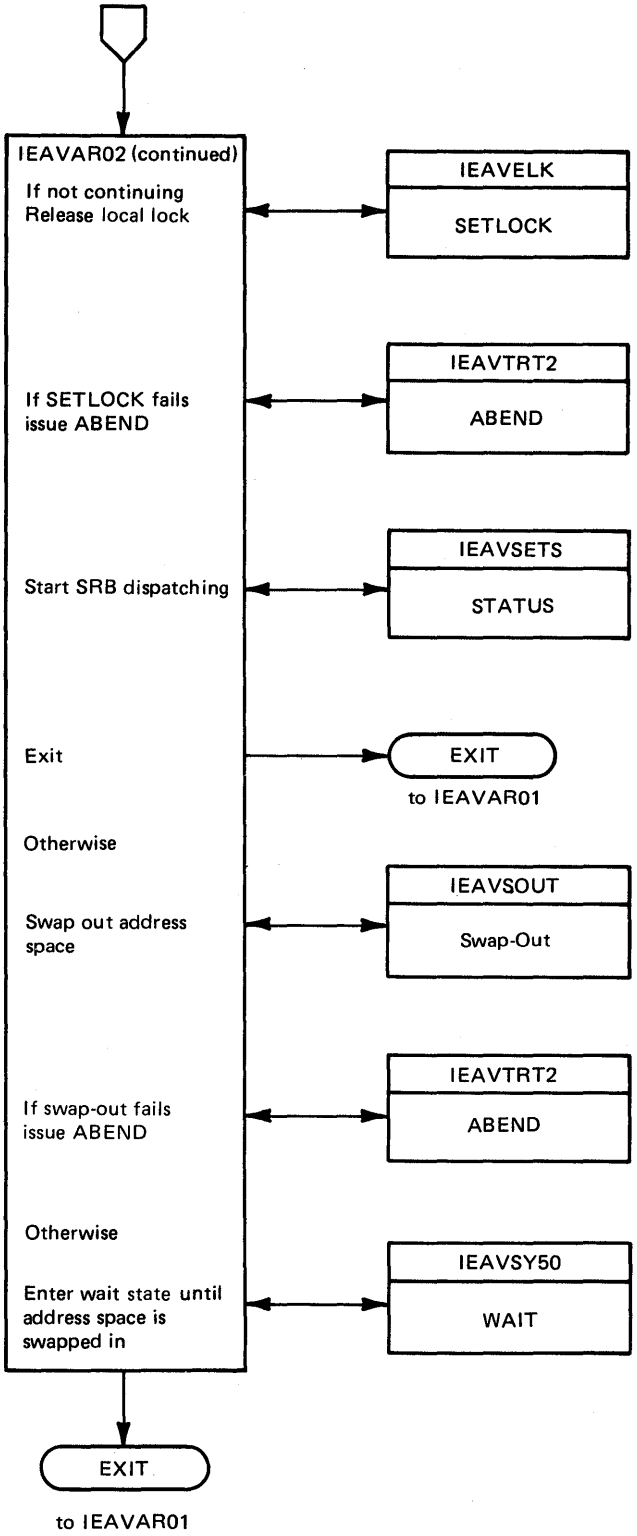


Figure 3-7. Region Control Task (RCT) Module Flow (Part 5 of 10)

RESTORE

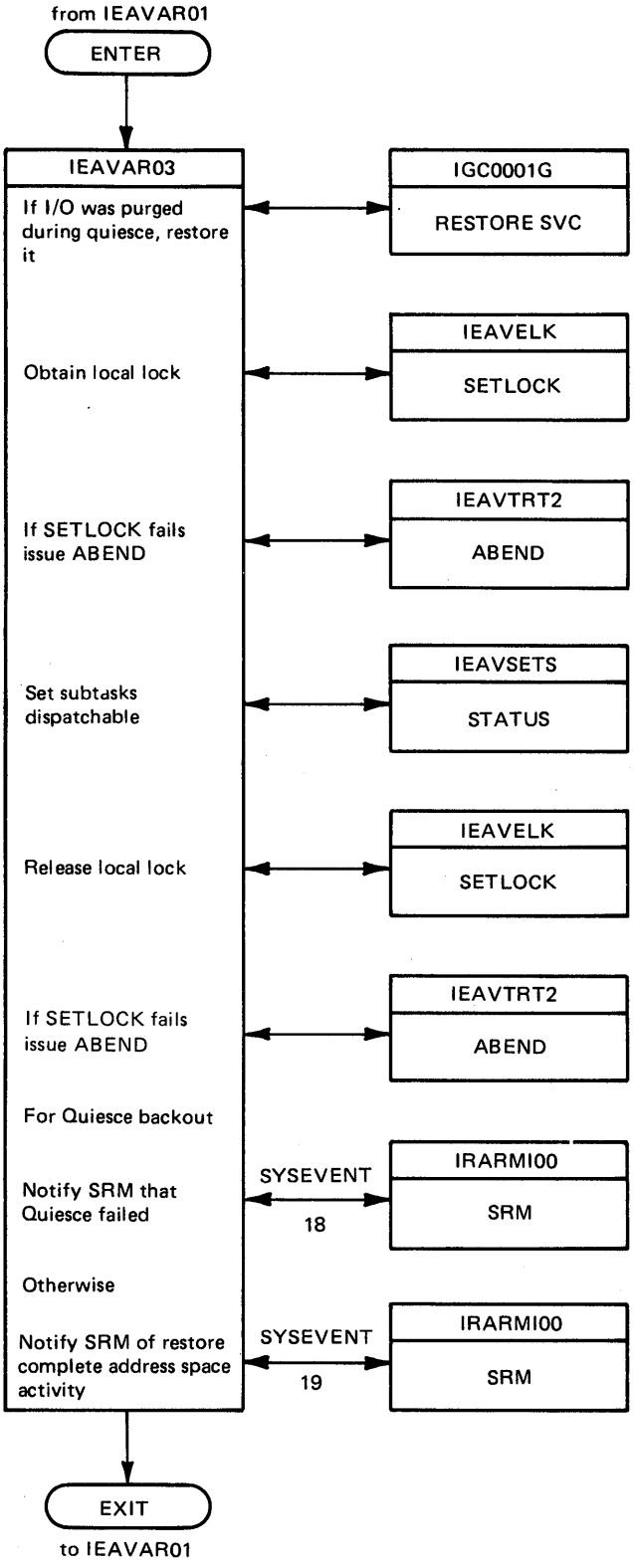


Figure 3-7. Region Control Task (RCT) Module Flow (Part 6 of 10)

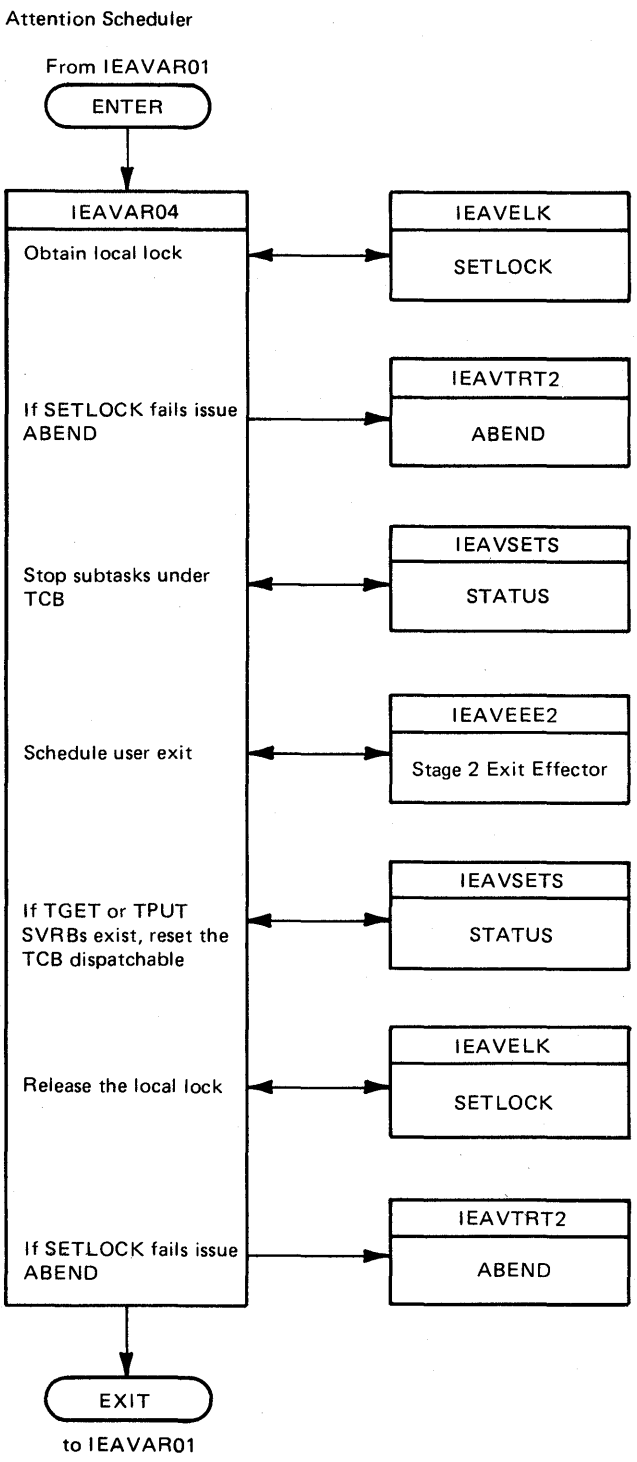


Figure 3-7. Region Control Task (RCT) Module Flow (Part 7 of 10)

Attention Exit Prolog and Epilog

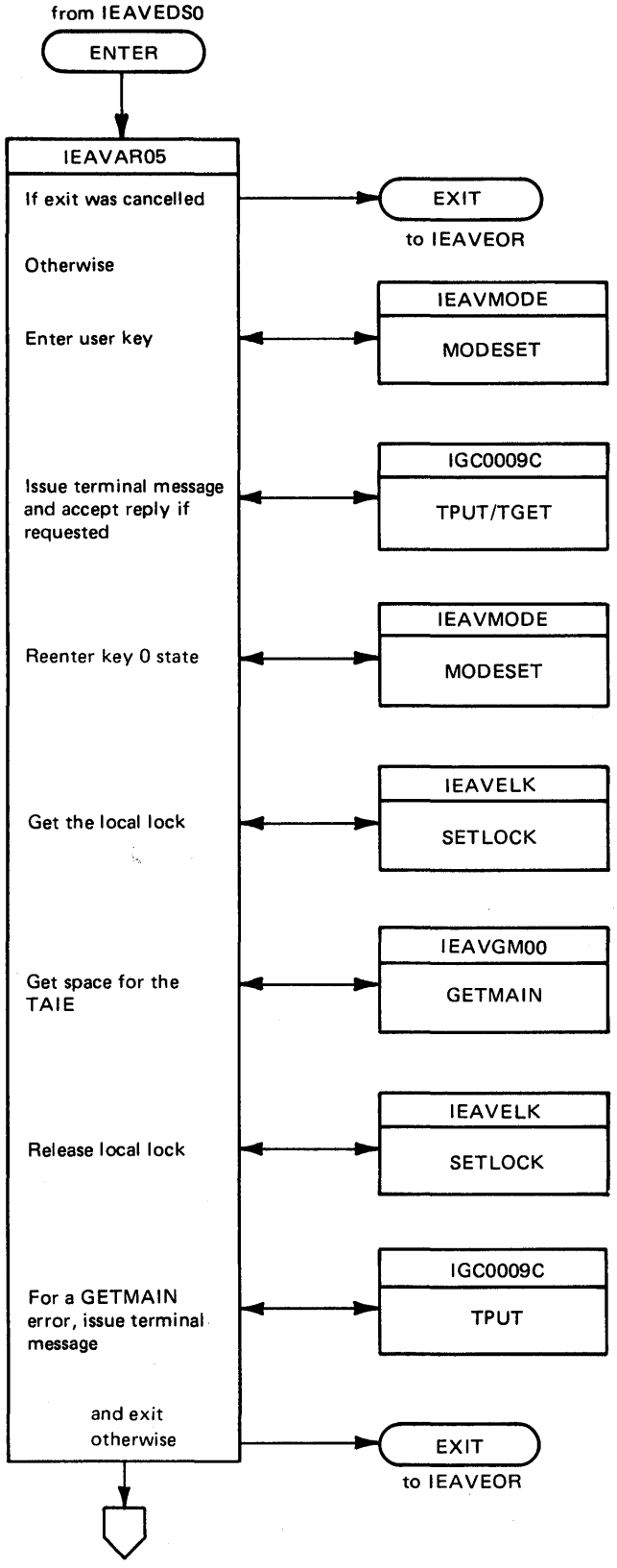


Figure 3-7. Region Control Task (RCT) Module Flow (Part 8 of 10)

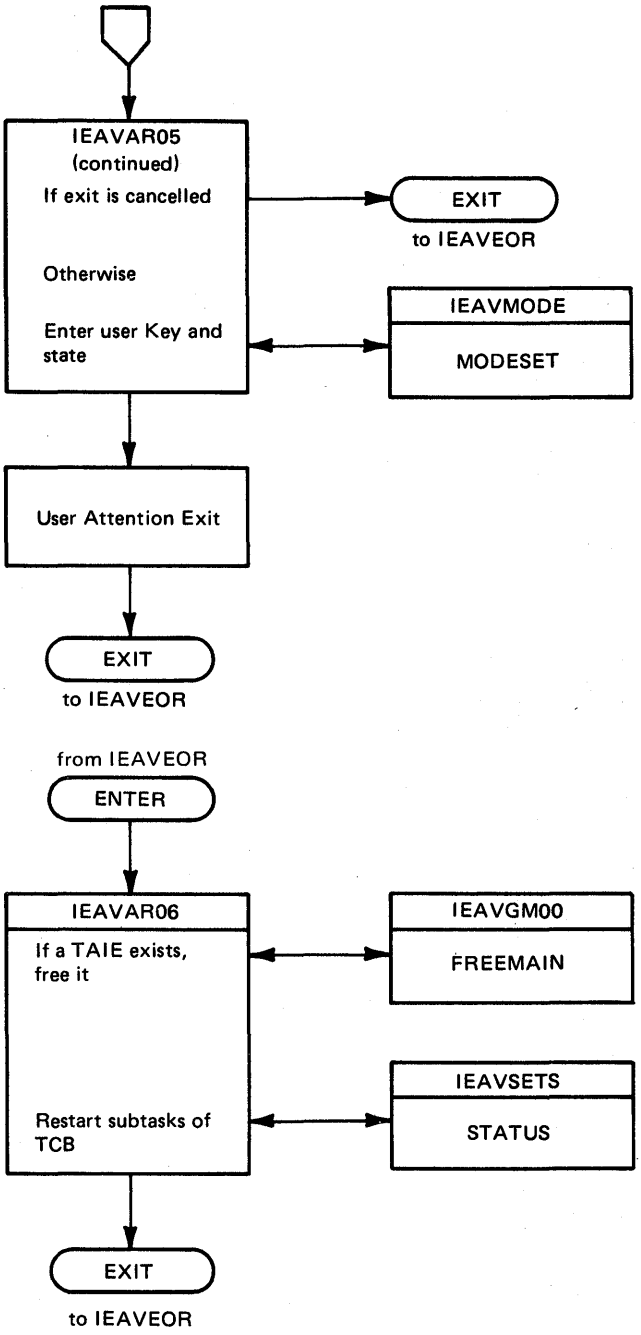


Figure 3-7. Region Control Task (RCT) Module Flow (Part 9 of 10)

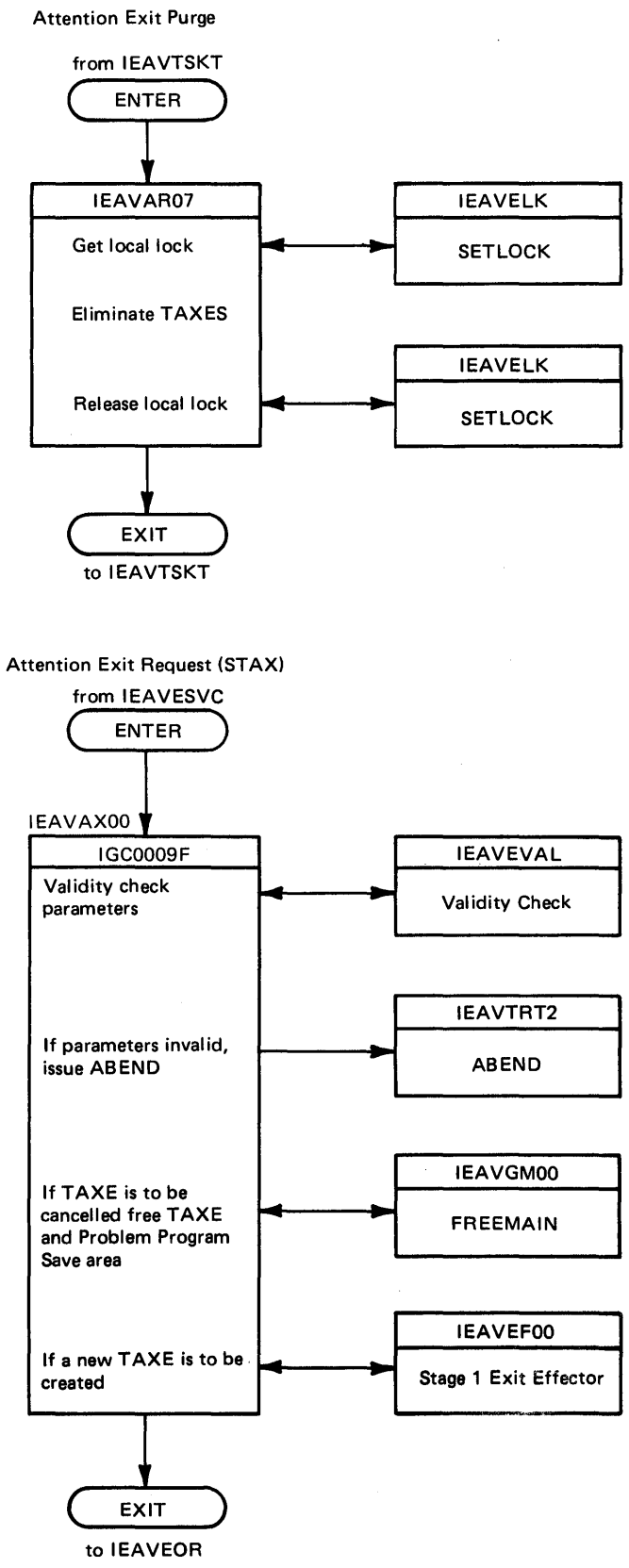


Figure 3-7. Region Control Task (RCT) Module Flow (Part 10 of 10)

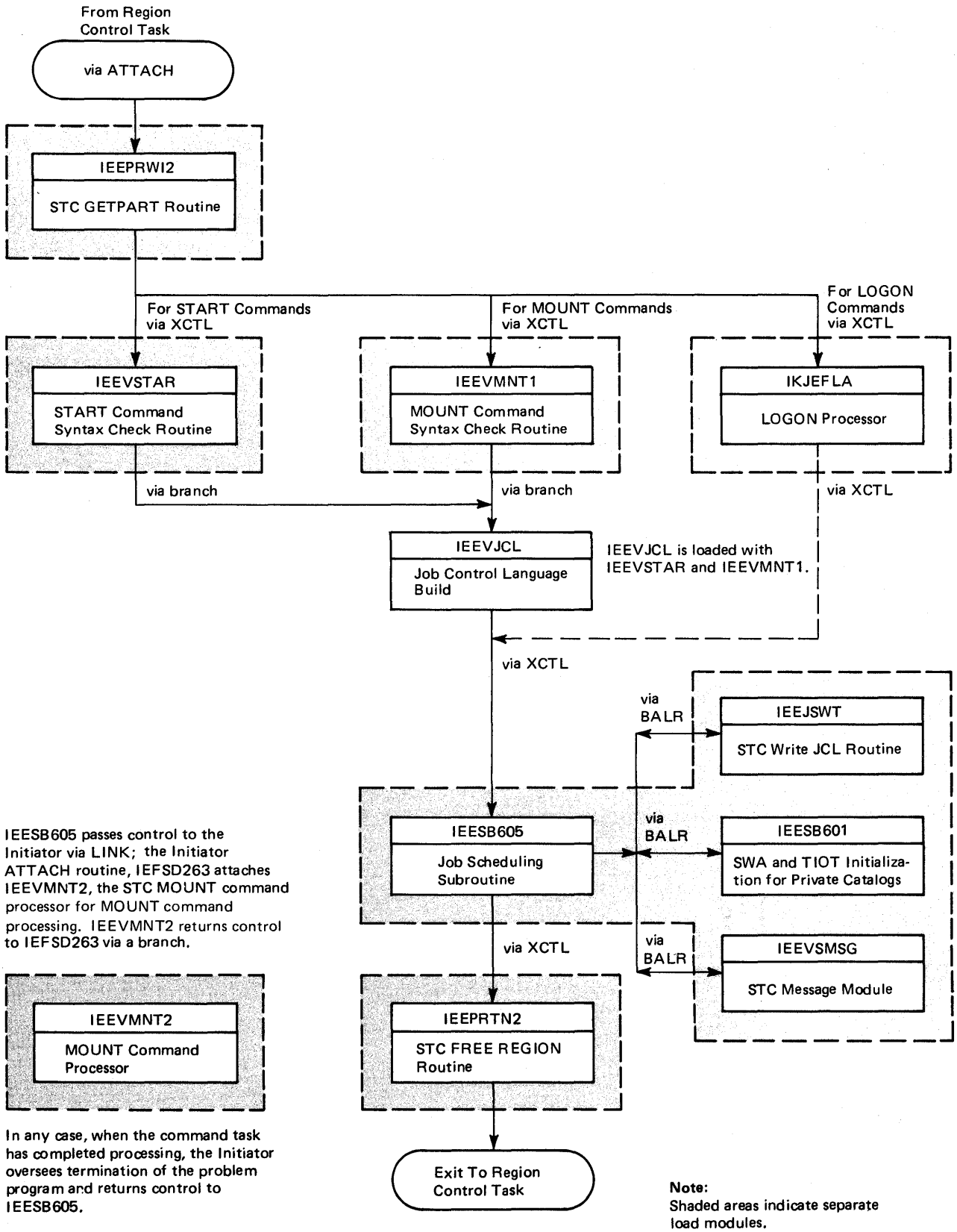


Figure 3-8. Started Task Control (STC) Module Flow (Part 1 of 2)

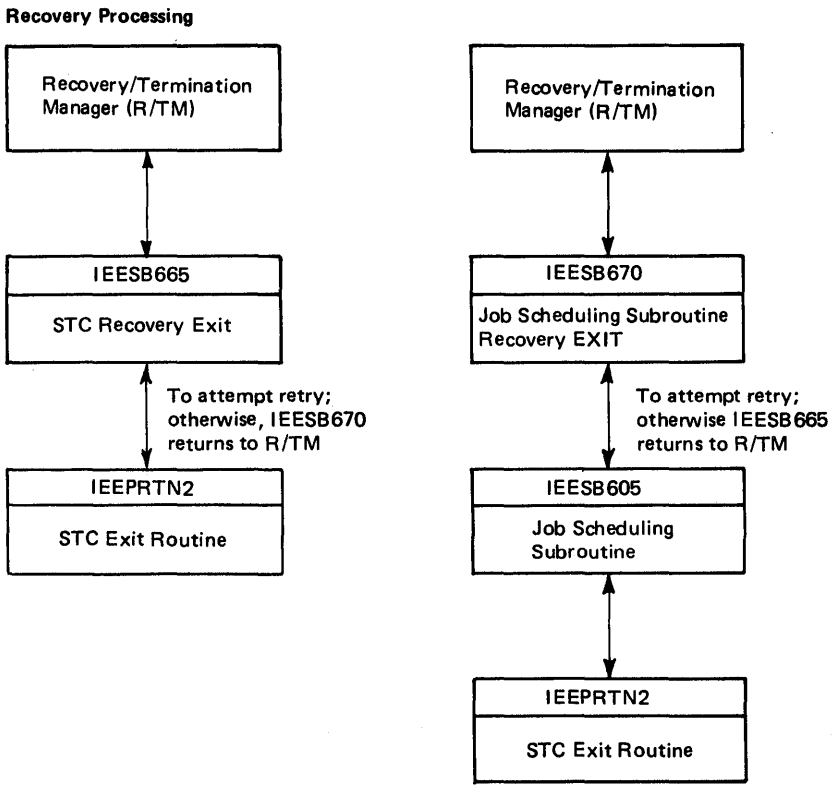


Figure 3-8. Started Task Control (STC) Module Flow (Part 2 of 2)

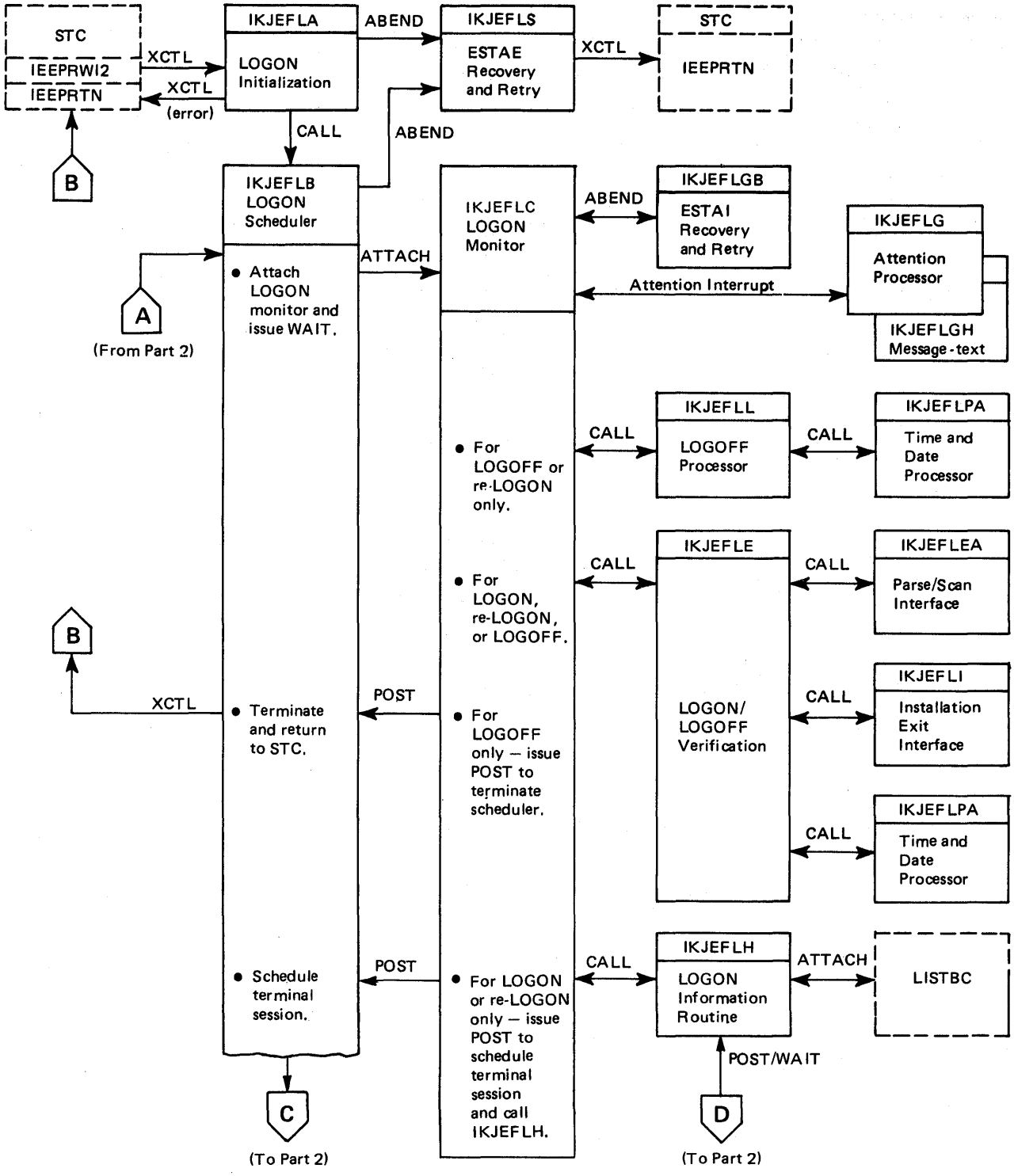


Figure 3-9. LOGON Scheduling Module Flow (Part 1 of 2)

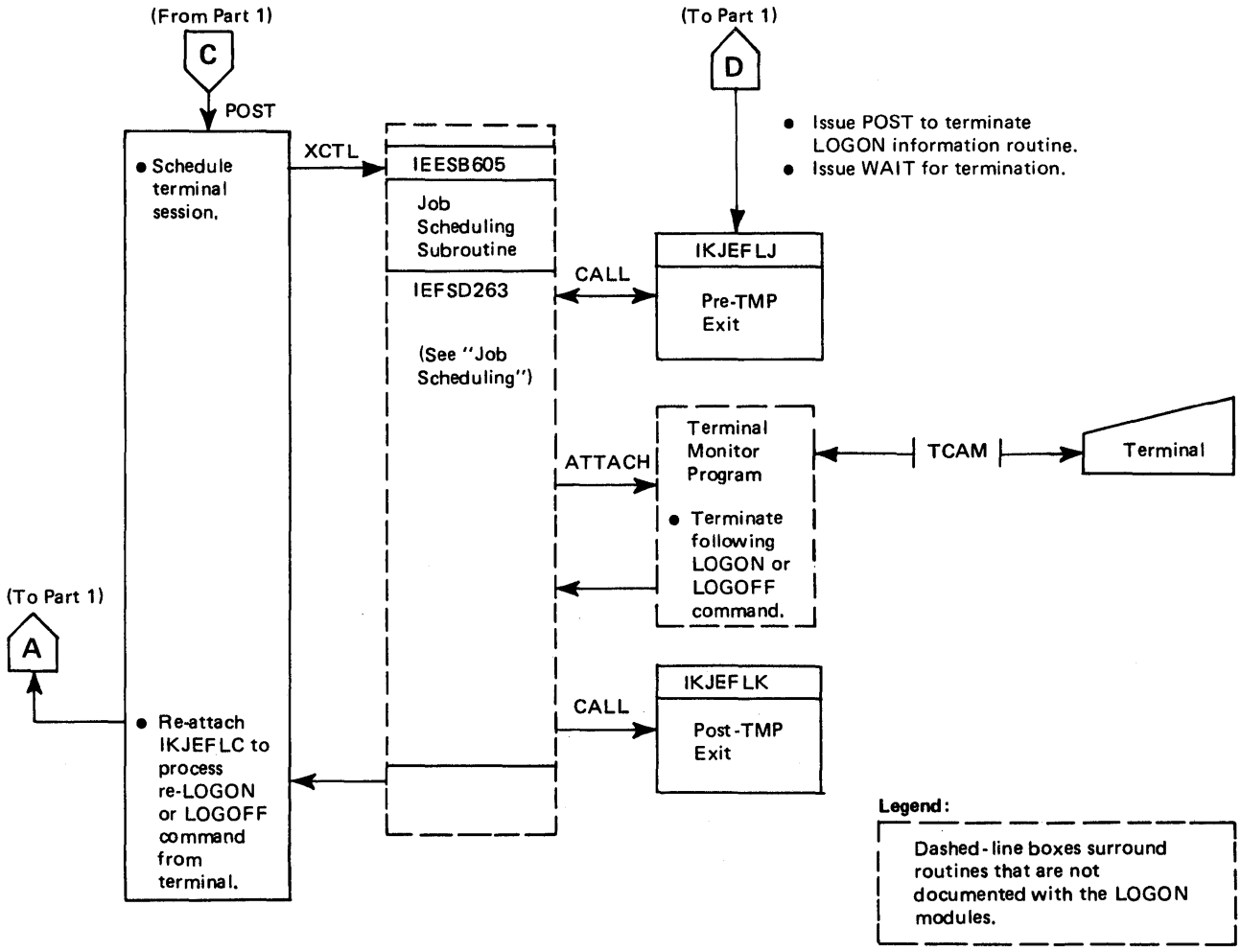


Figure 3-9. LOGON Scheduling Module Flow (Part 2 of 2)

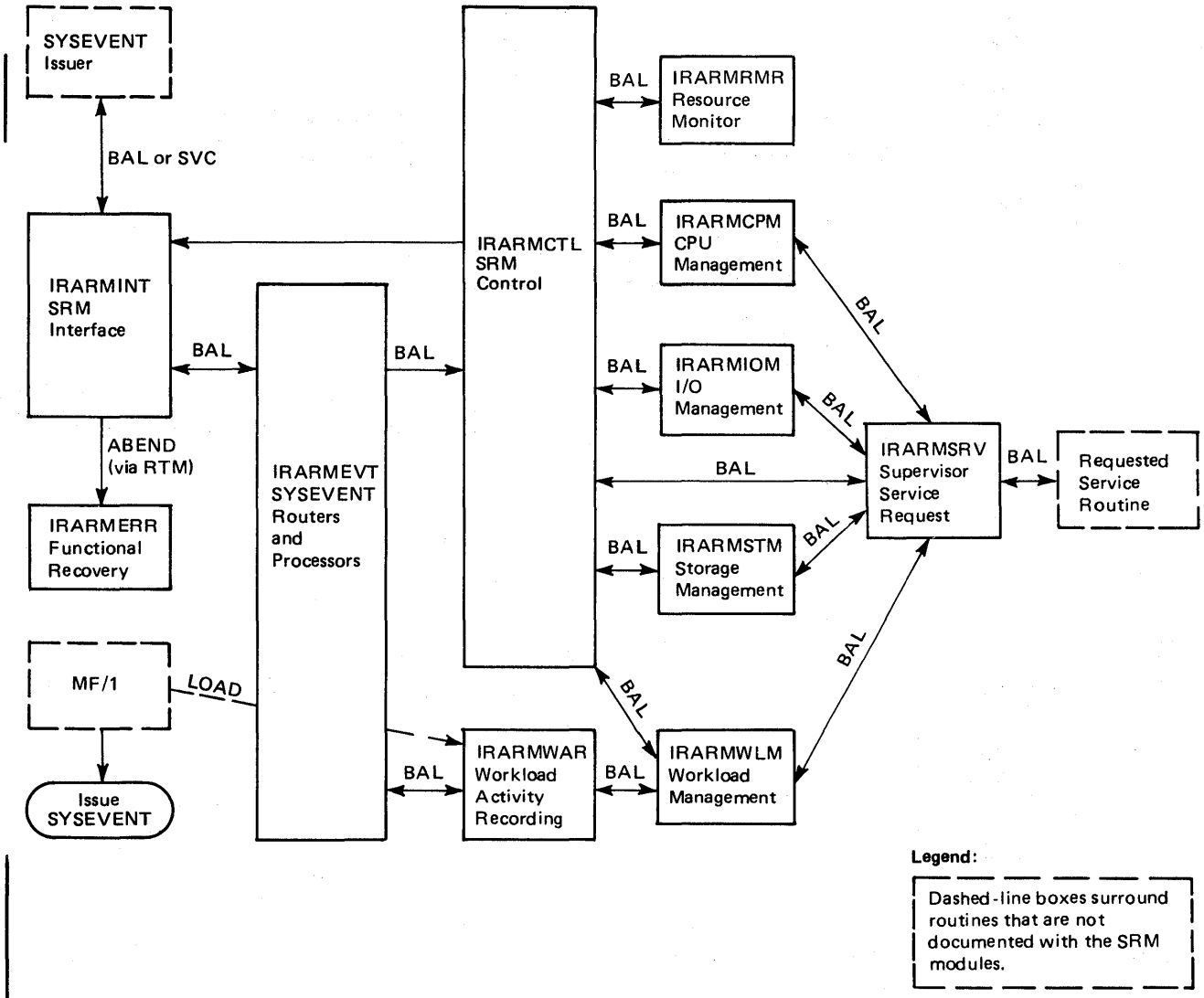


Figure 3-10. System Resources Manager (SRM) Mainline Processing Module Flow

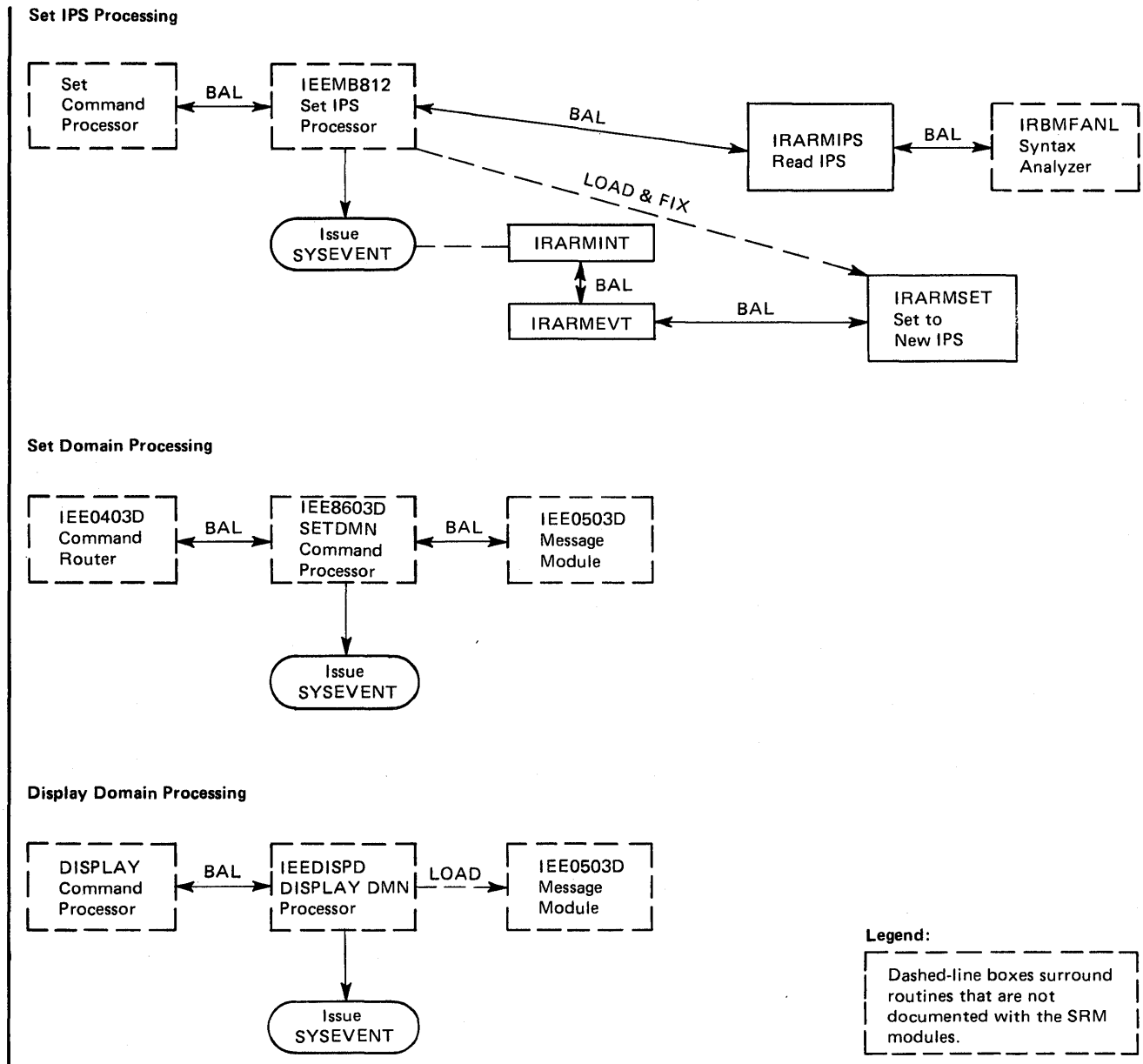


Figure 3-10a. System Resources Manager (SRM) Command Processing Module Flow

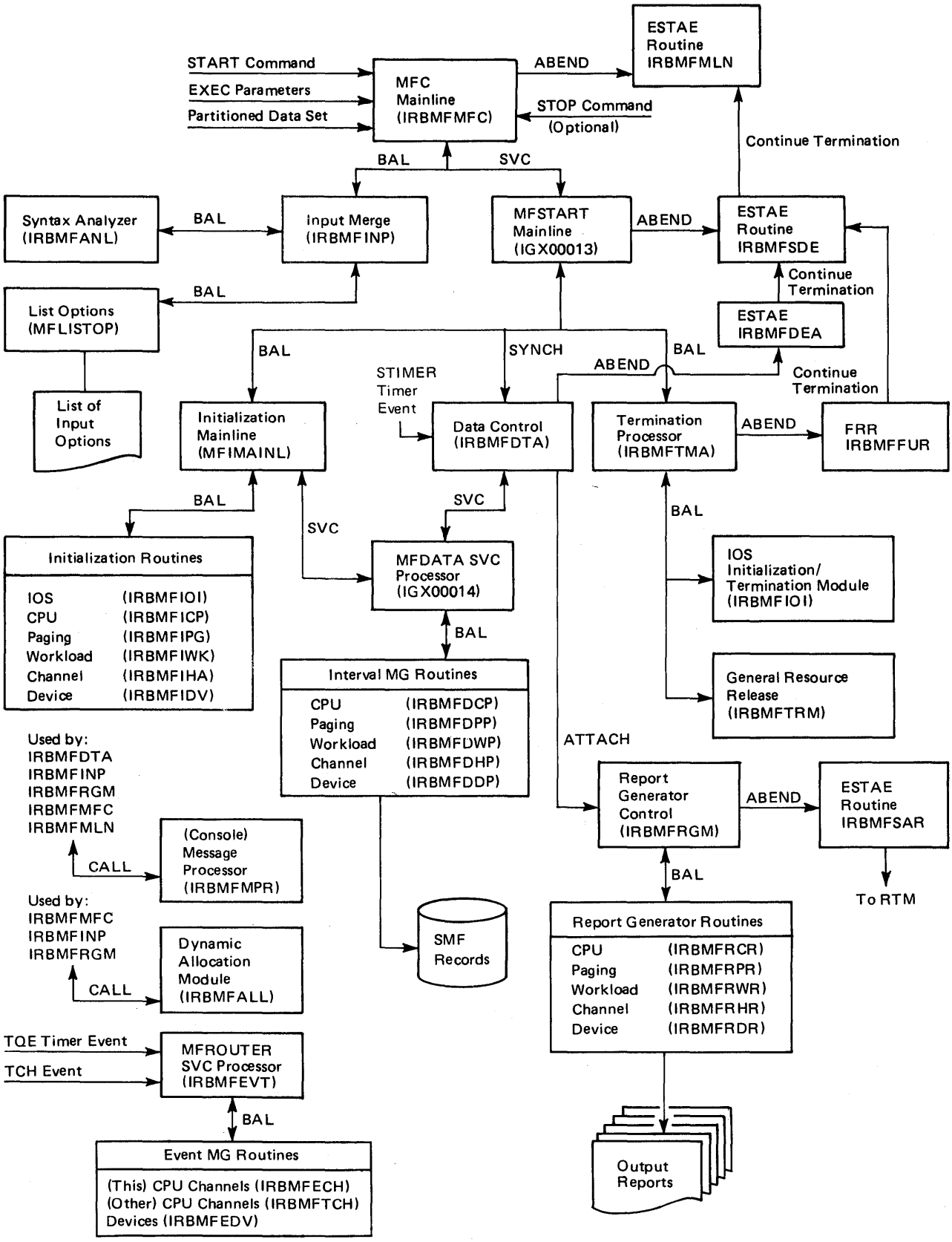


Figure 3-11. System Activity Measurement Facility (MF/1) Module Flow

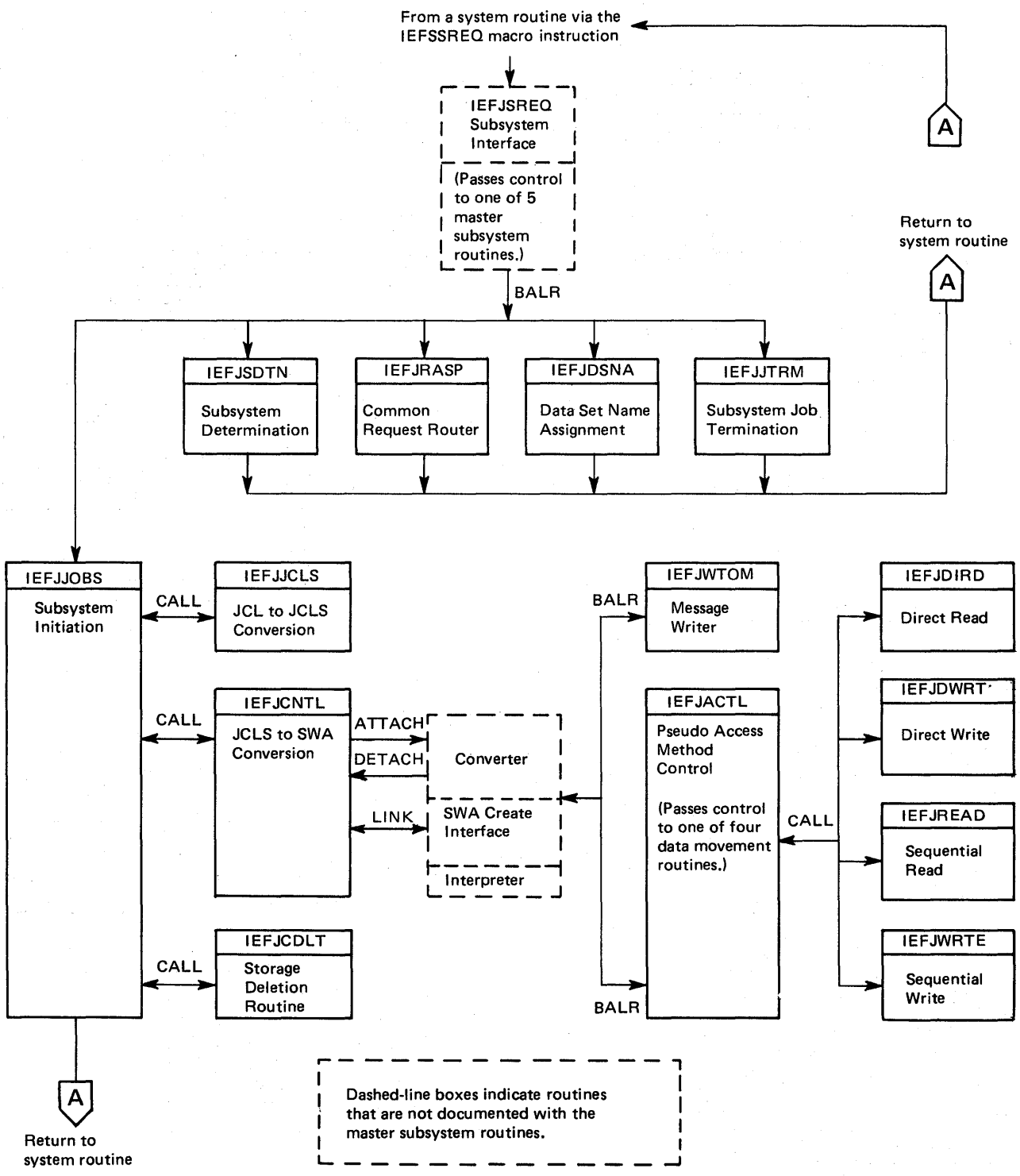


Figure 3-12. Master Subsystem Module Flow

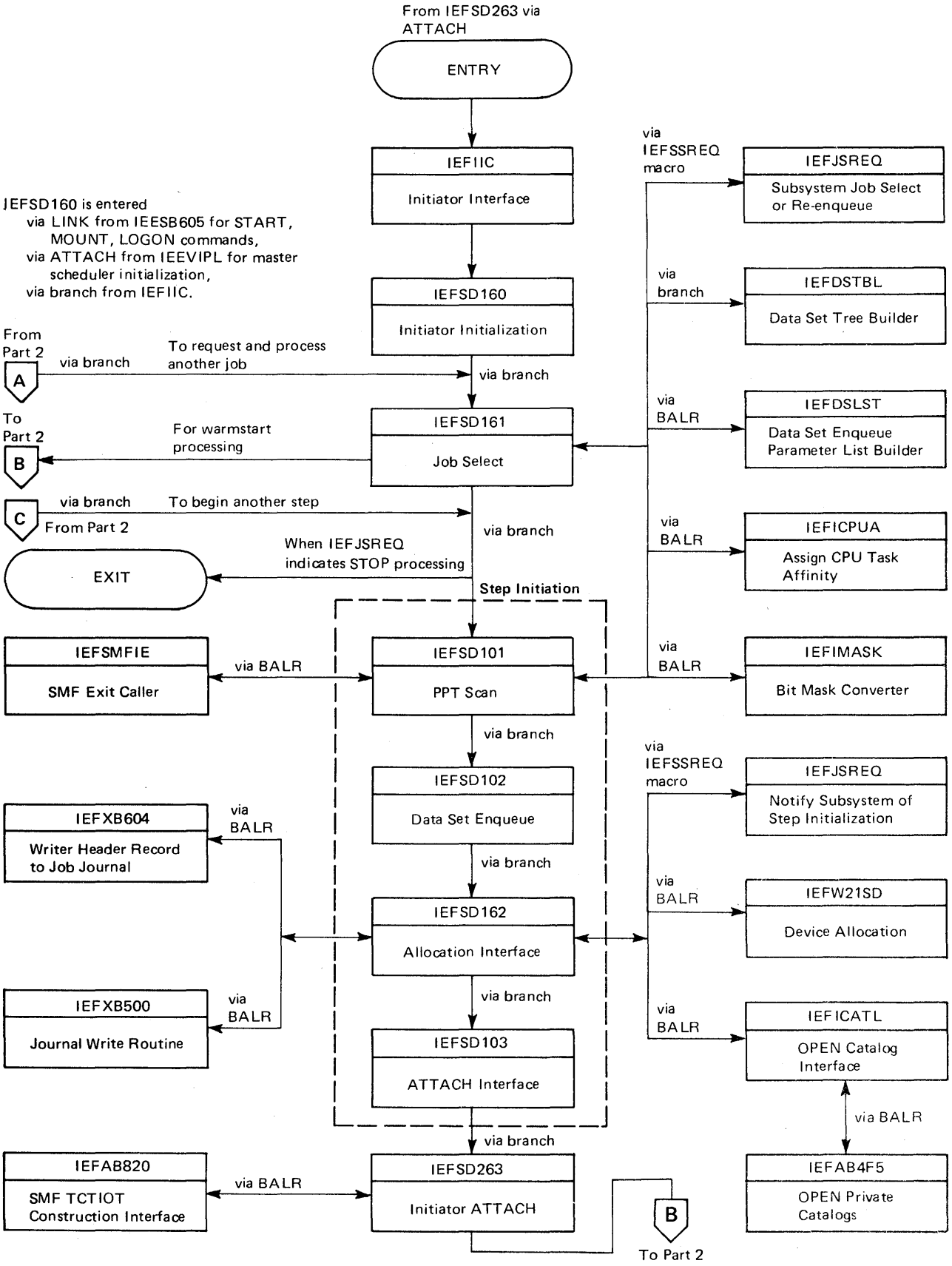


Figure 3-13. Initiator/Terminator Module Flow (Part 1 of 2)

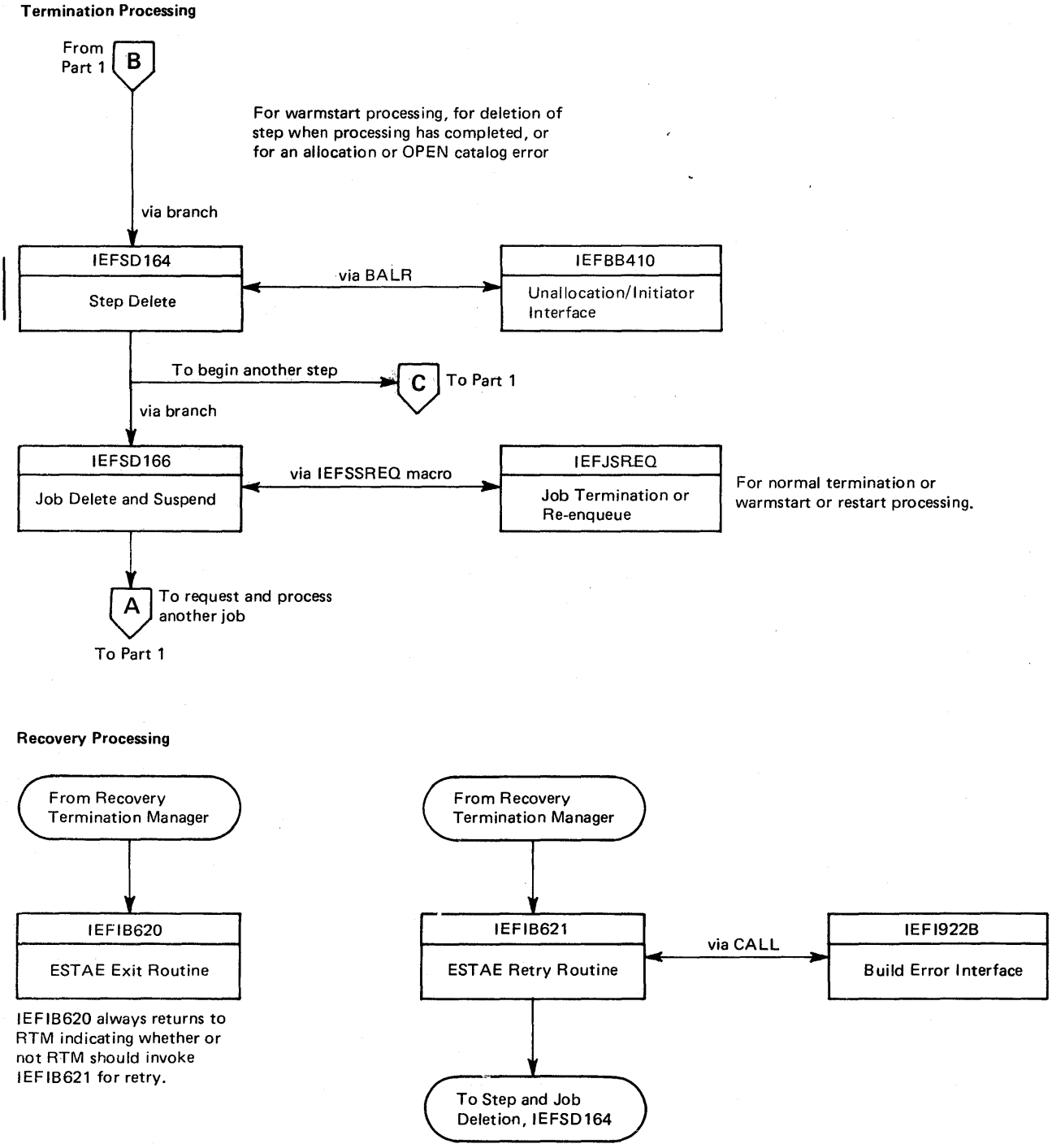


Figure 3-13. Initiator/Terminator Module Flow (Part 2 of 2)

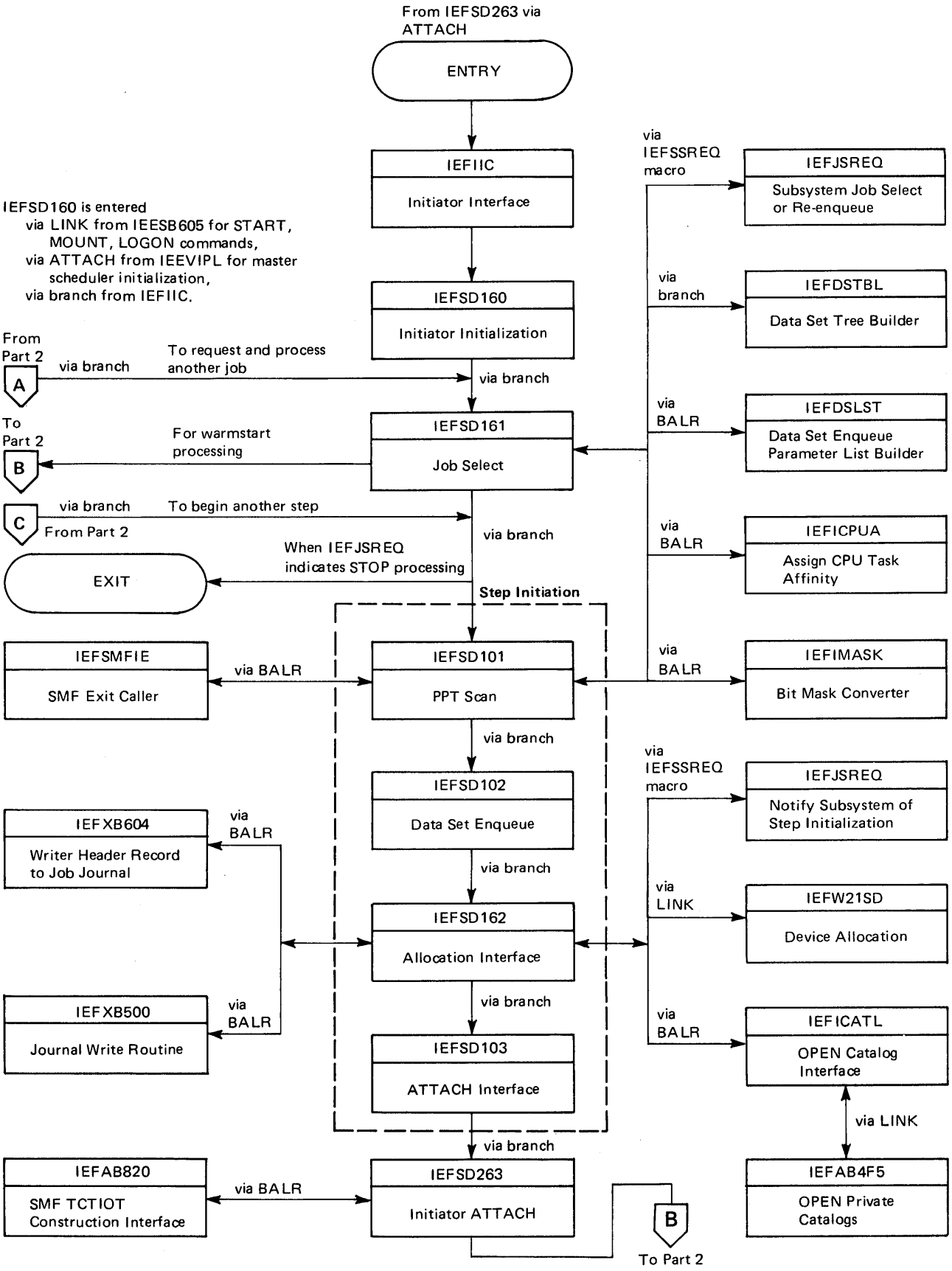


Figure 3-13. Initiator/Terminator Module Flow (Part 1 of 2)

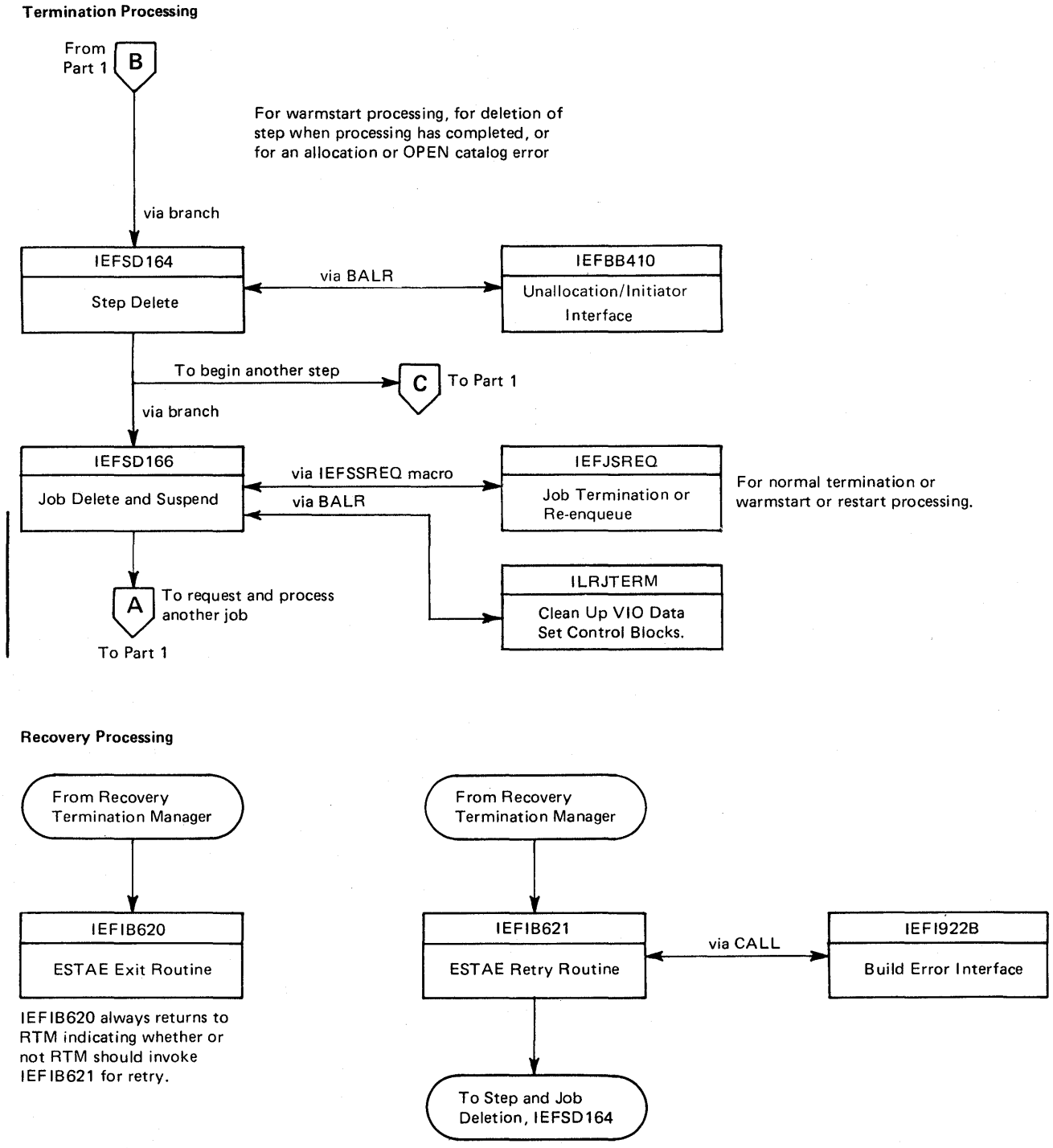


Figure 3-13. Initiator/Terminator Module Flow (Part 2 of 2)

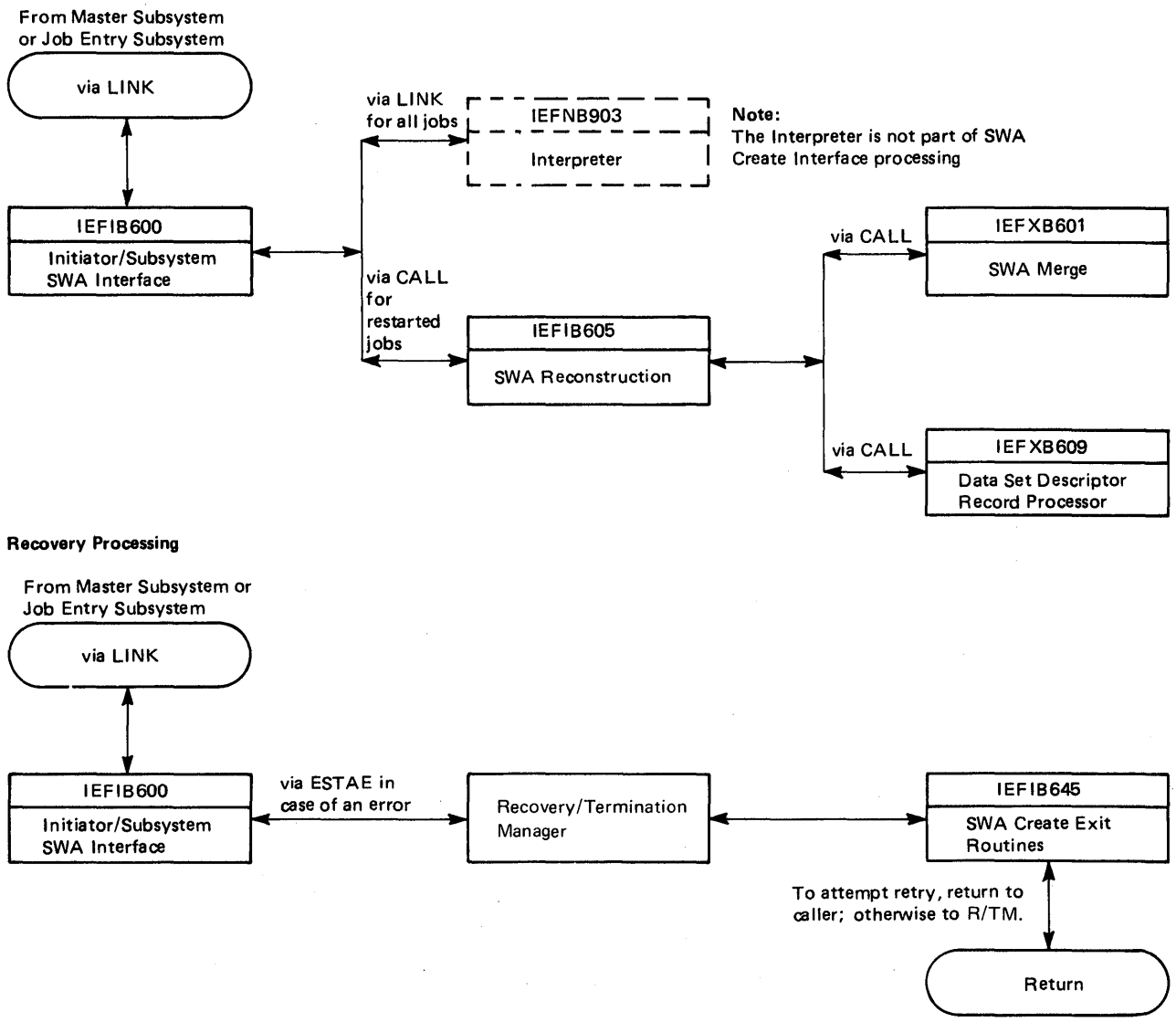


Figure 3-14. SWA Create Interface Module Flow

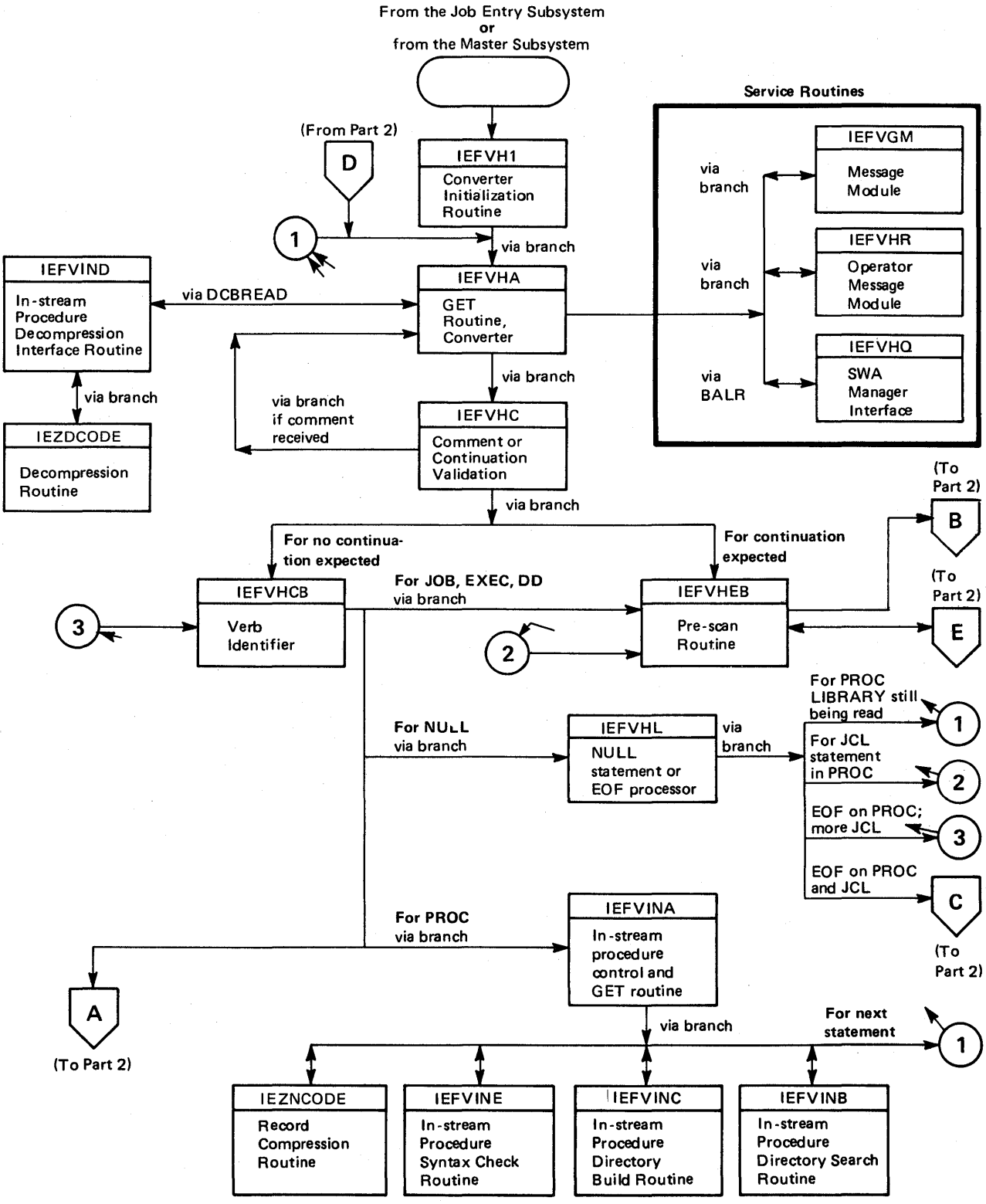


Figure 3-15. Converter Module Flow (Part 1 of 2)

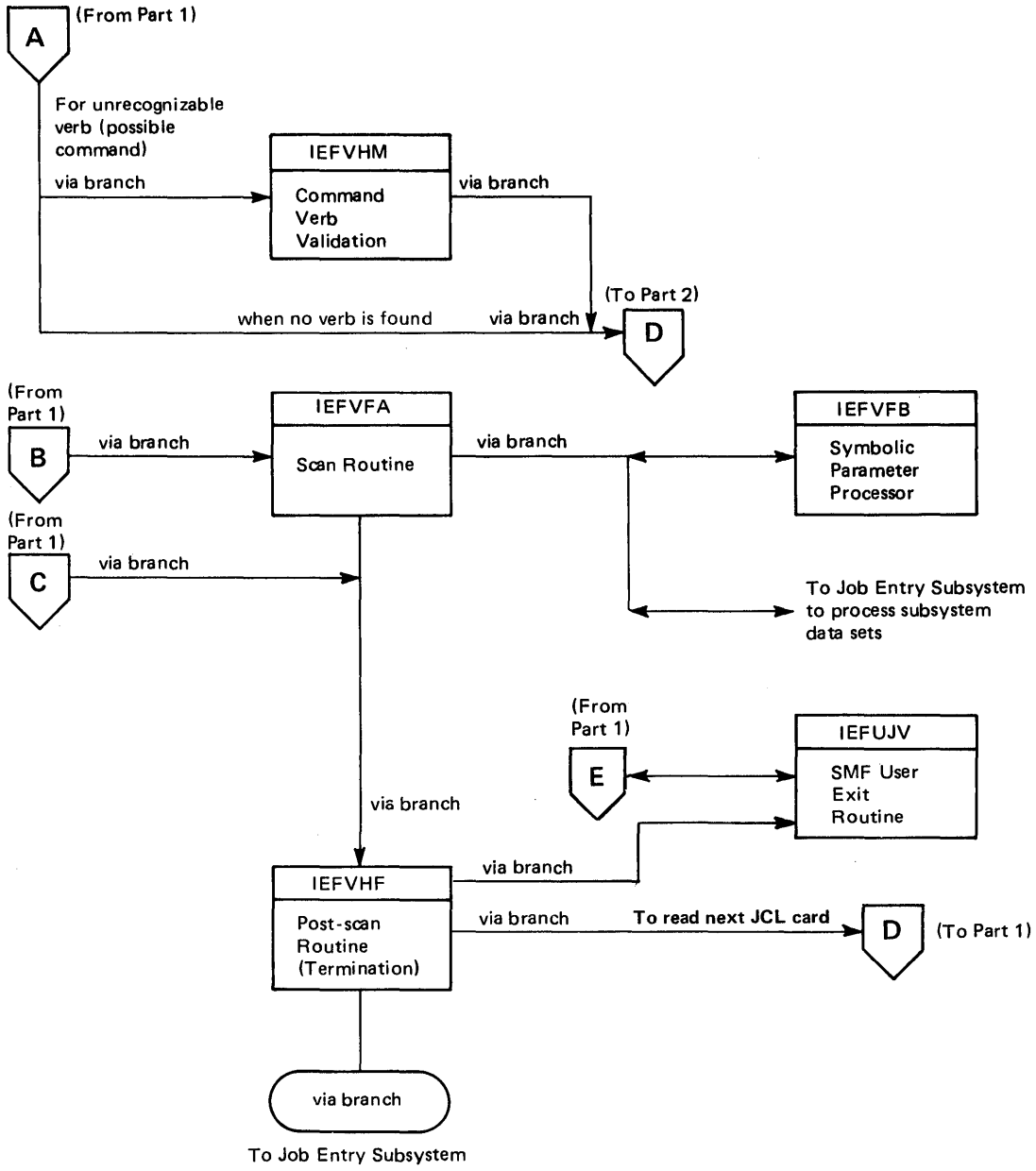


Figure 3-15. Converter Module Flow (Part 2 of 2)

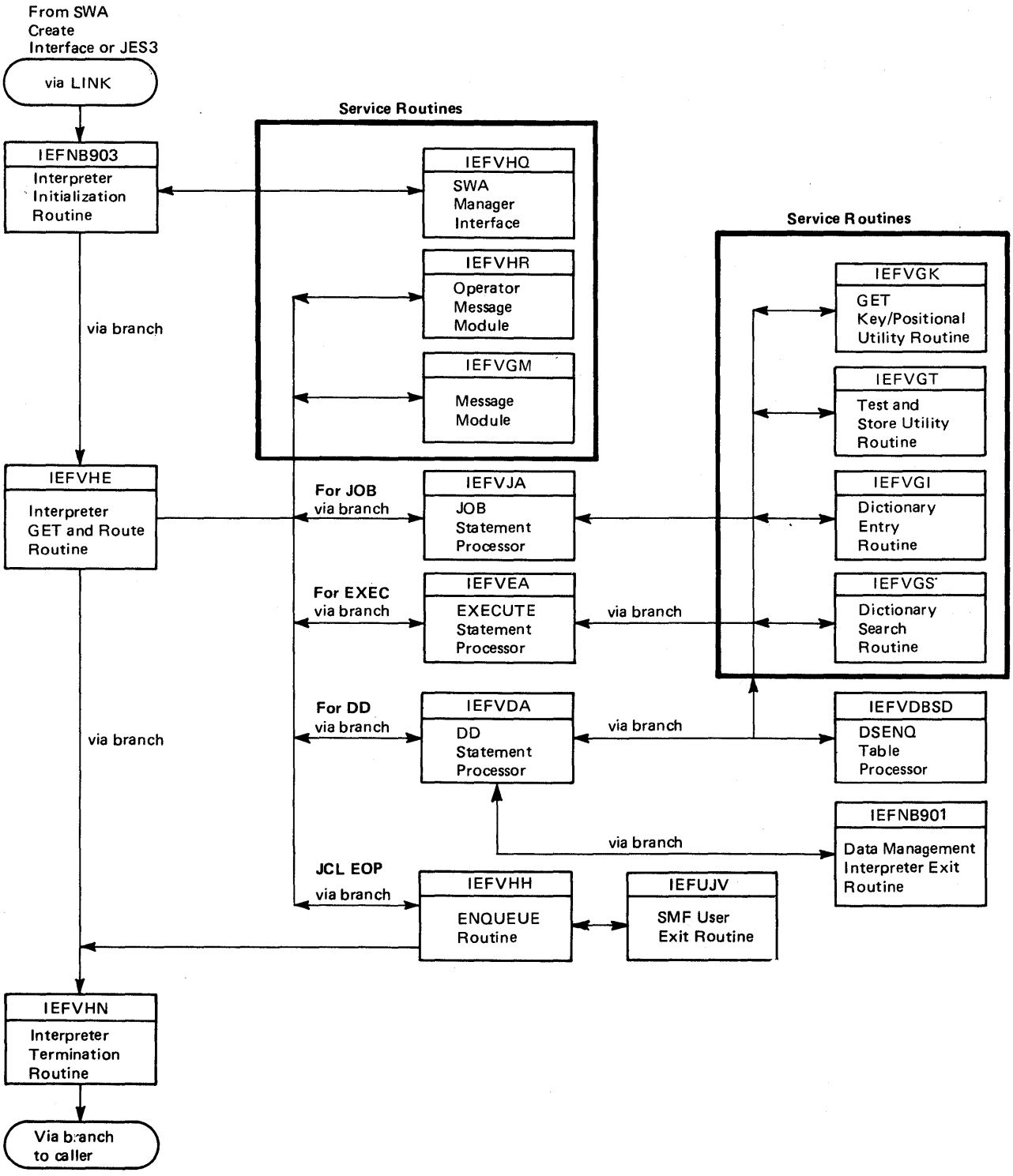
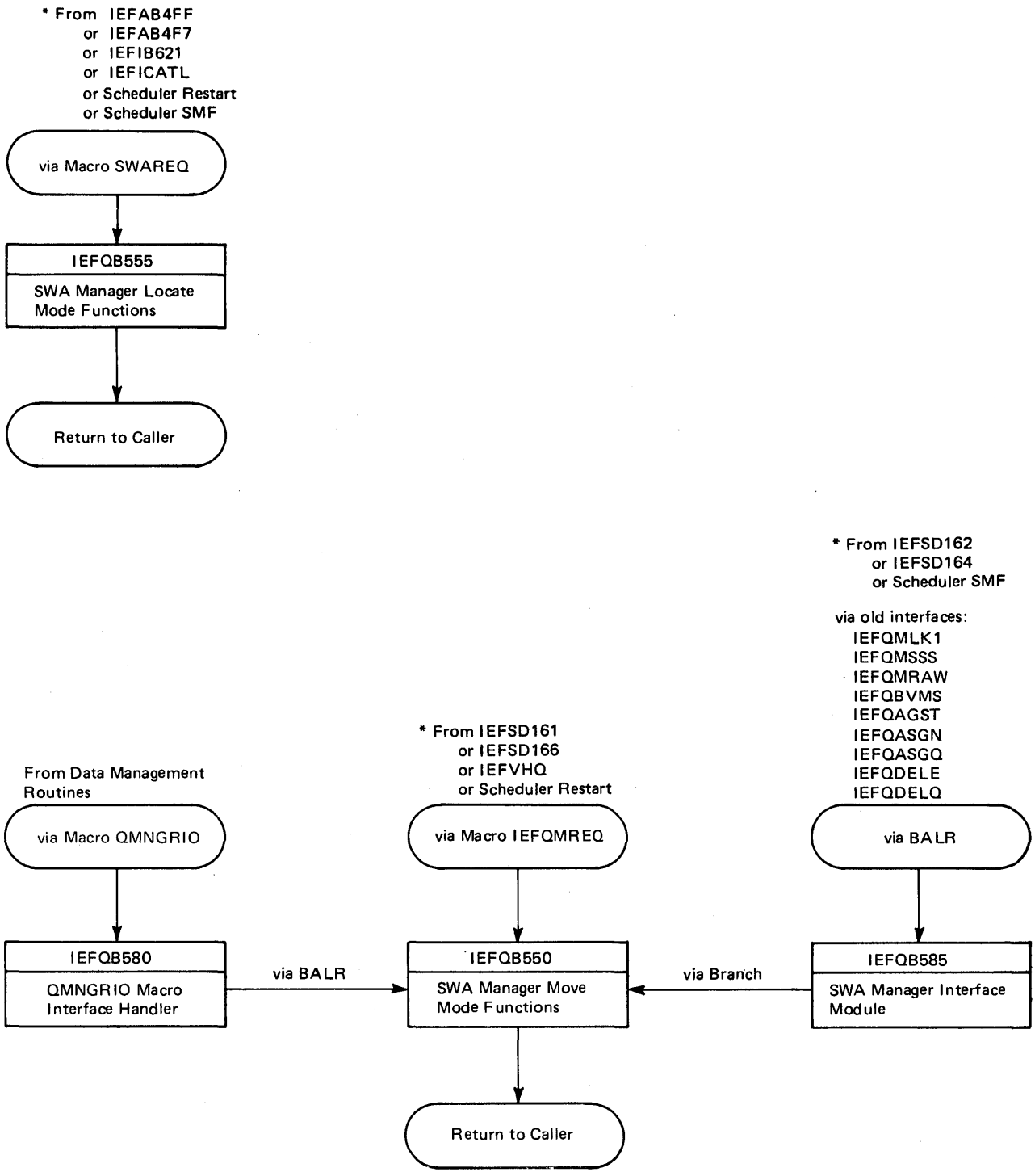


Figure 3-16. Interpreter Module Flow

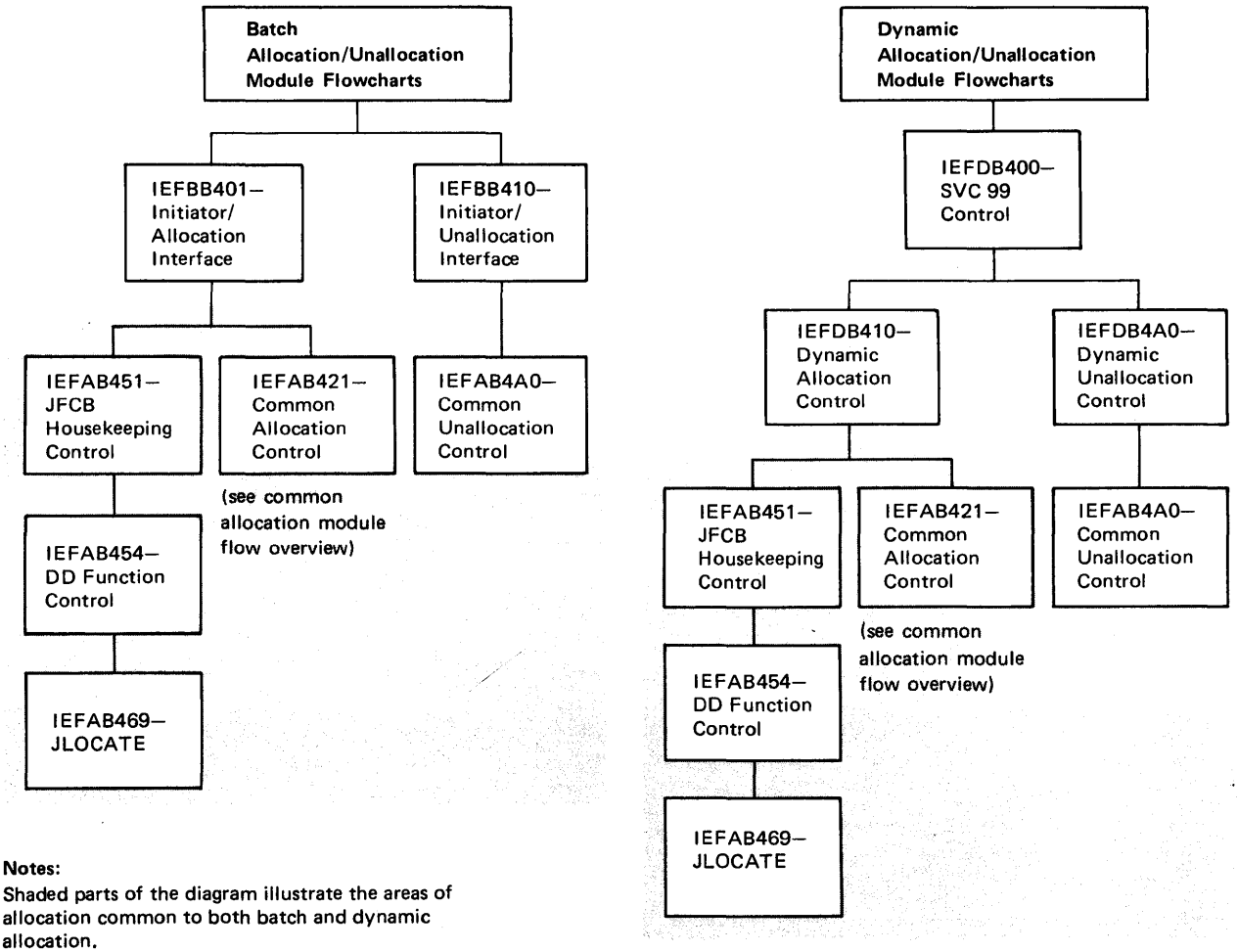


Note:
* This list is not inclusive; it represents some scheduler routines and components that frequently use SWA manager routines.

Figure 3-17. SWA Manager Module Flow

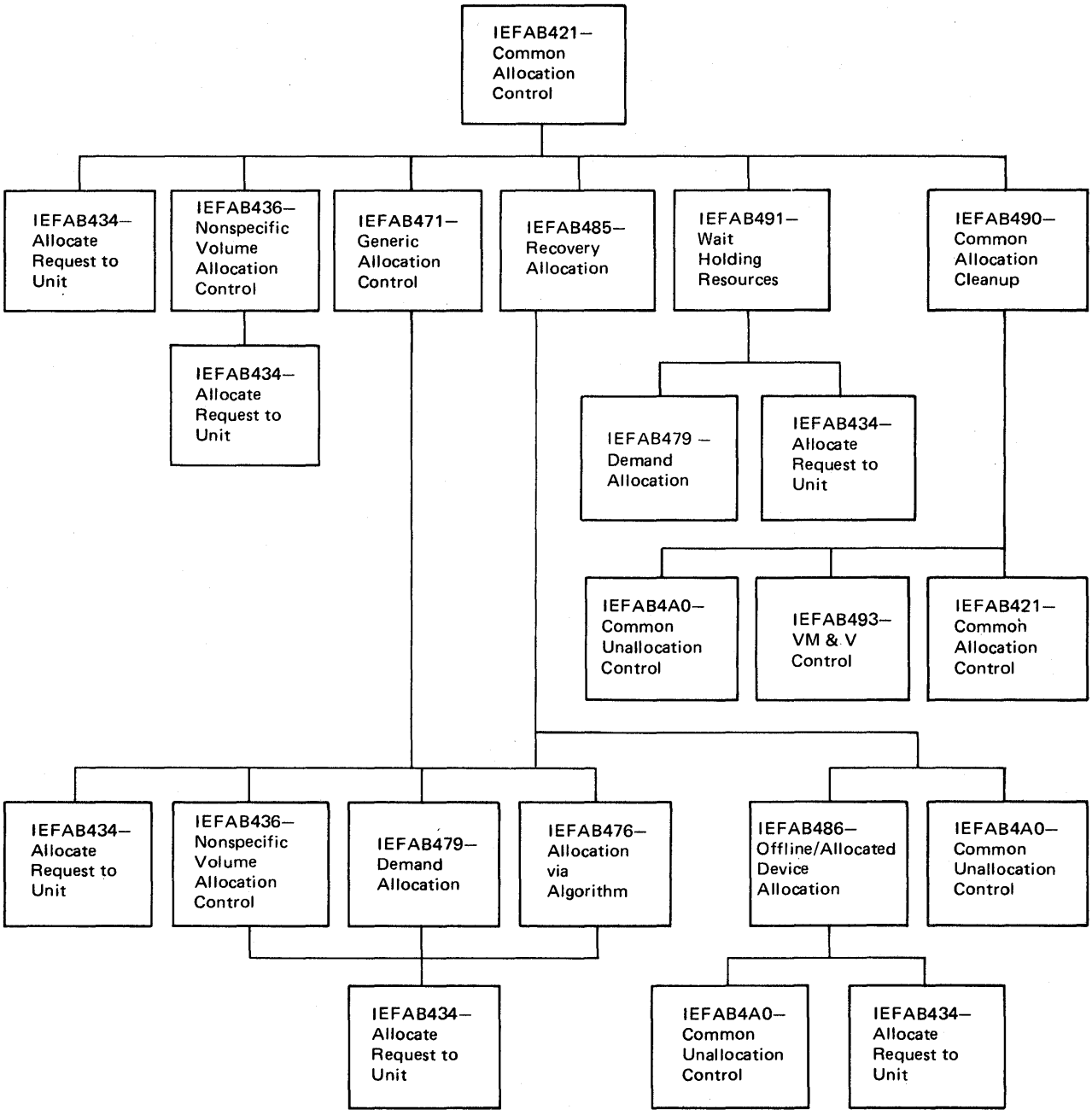
Allocation/Unallocation Module Flow

Module flow for Allocation/Unallocation is illustrated in several module flowcharts. Each flowchart is titled according to the major module it represents; they are arranged in alphabetic order, according to the module name. Two overviews are included to illustrate the processing hierarchy of the module flowcharts: an overview of batch and dynamic module flowcharts; and an overview of common allocation module flowcharts.



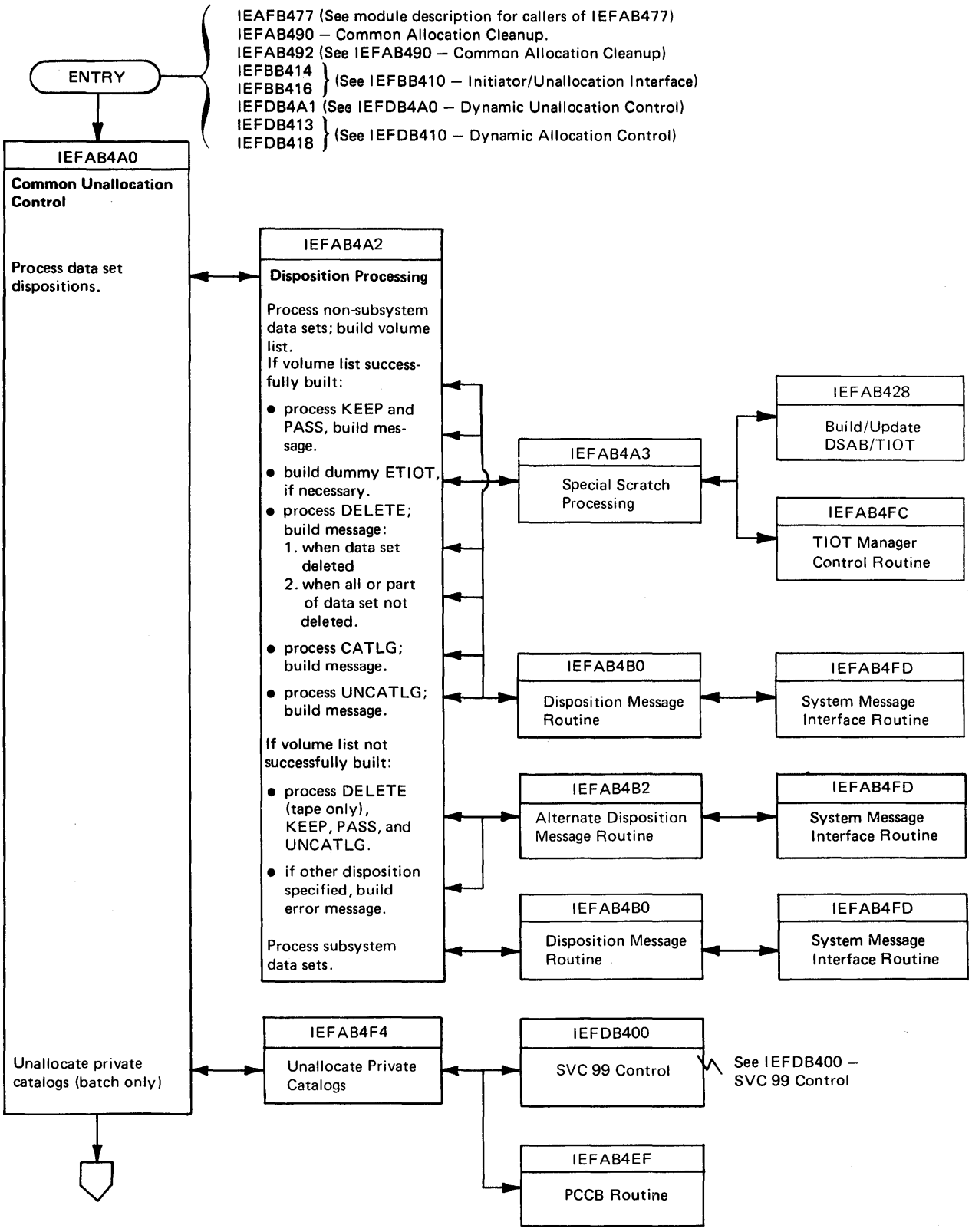
Notes:
Shaded parts of the diagram illustrate the areas of allocation common to both batch and dynamic allocation.
The module flowcharts are arranged in alphabetic order. This overview is intended to illustrate the hierarchy of the flowcharts; it does not indicate calling sequence between modules.

Figure 3-18. Batch and Dynamic Allocation/Unallocation Module Flow Overview



Note: The module flowcharts are arranged in alphabetic order. This overview is intended to illustrate the hierarchy of the flowcharts; it does not indicate calling sequence between modules.

Figure 3-19. Common Allocation Module Flow Overview



Note: All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

Figure 3-20. IEFAB4A0 – Common Unallocation Control (Part 1 of 2)

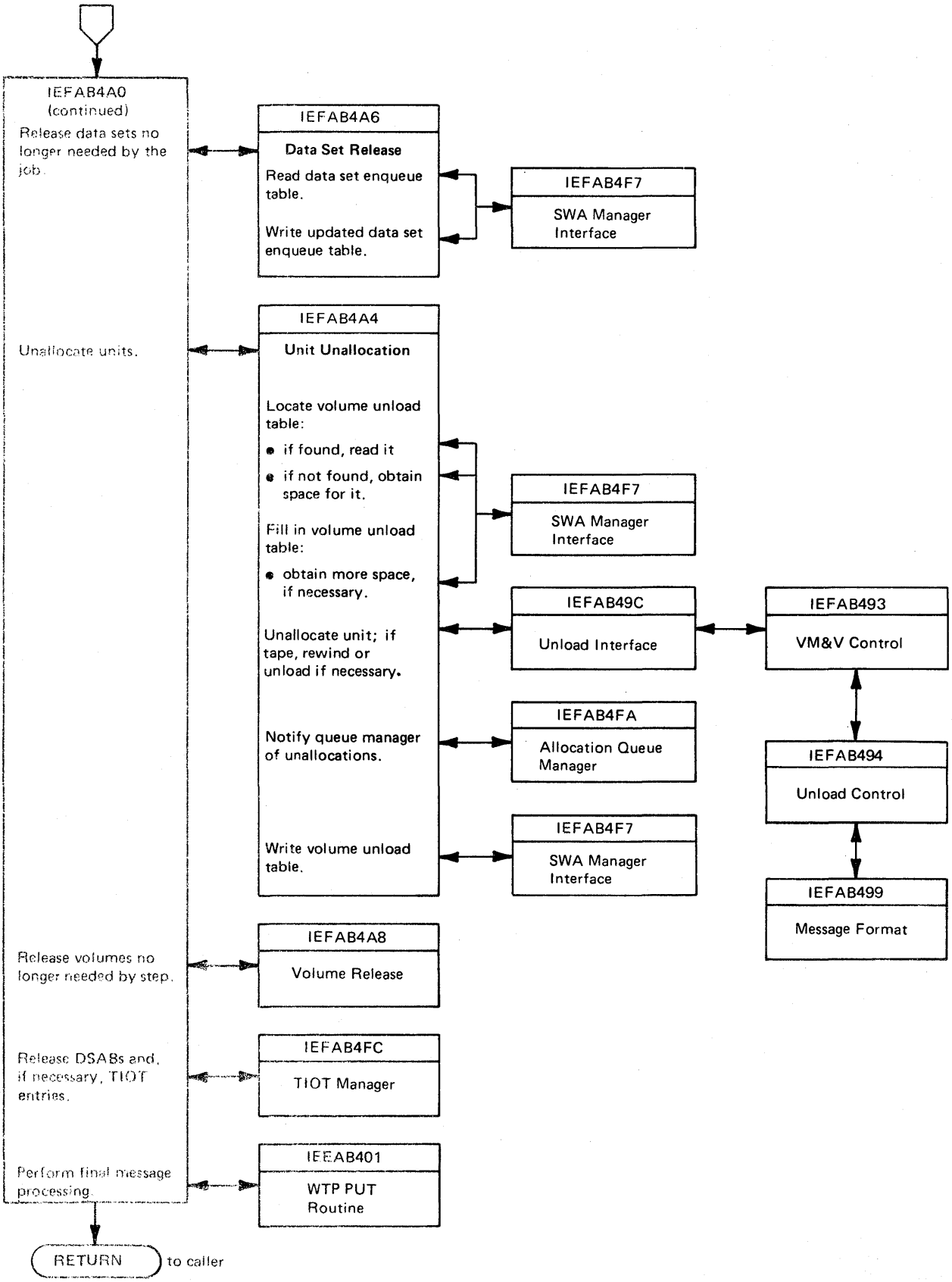


Figure 3-20. IEFAB4A0 – Common Unallocation Control (Part 2 of 2)

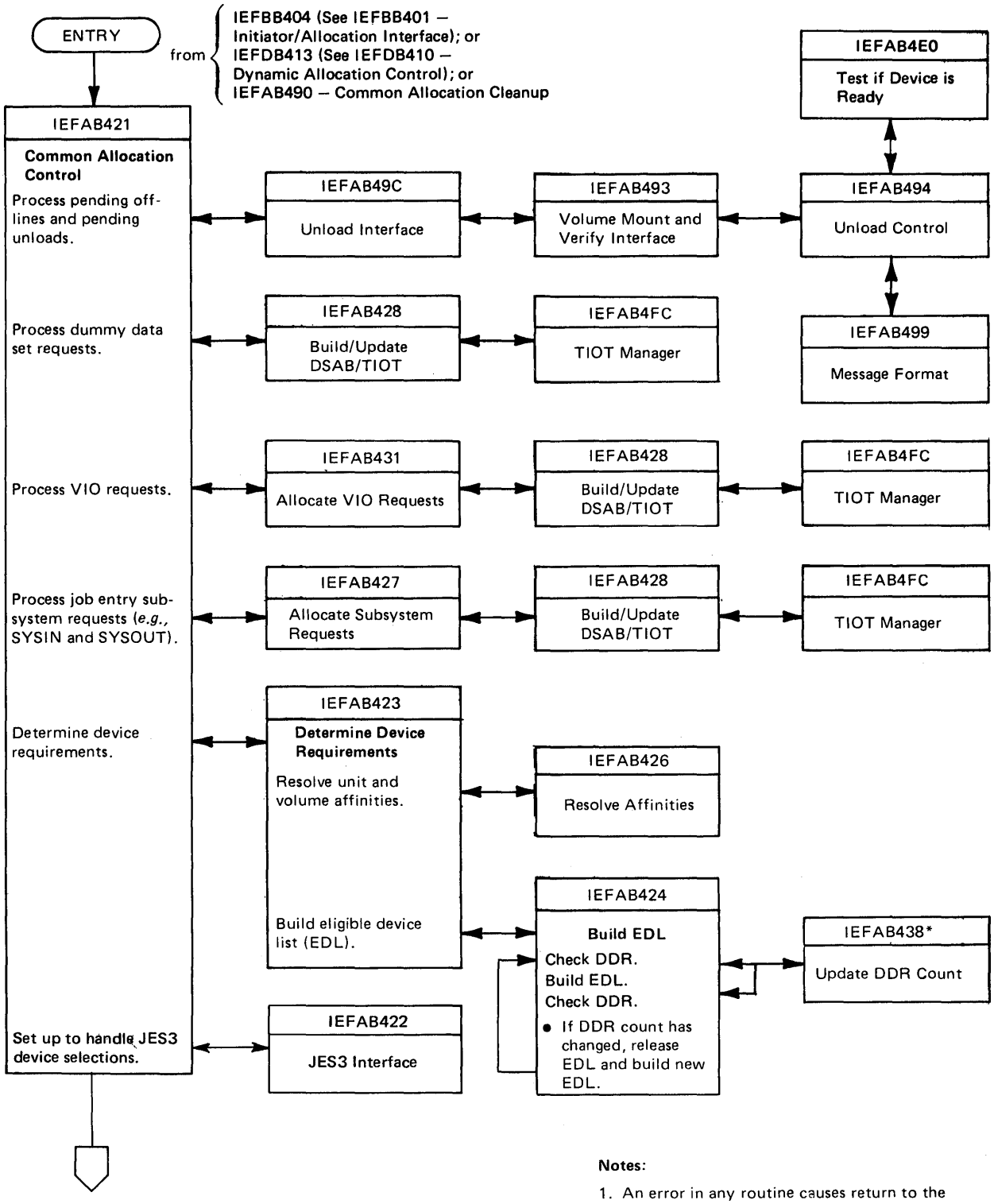


Figure 3-21. IEFAB421 – Common Allocation Control (Part 1 of 2)

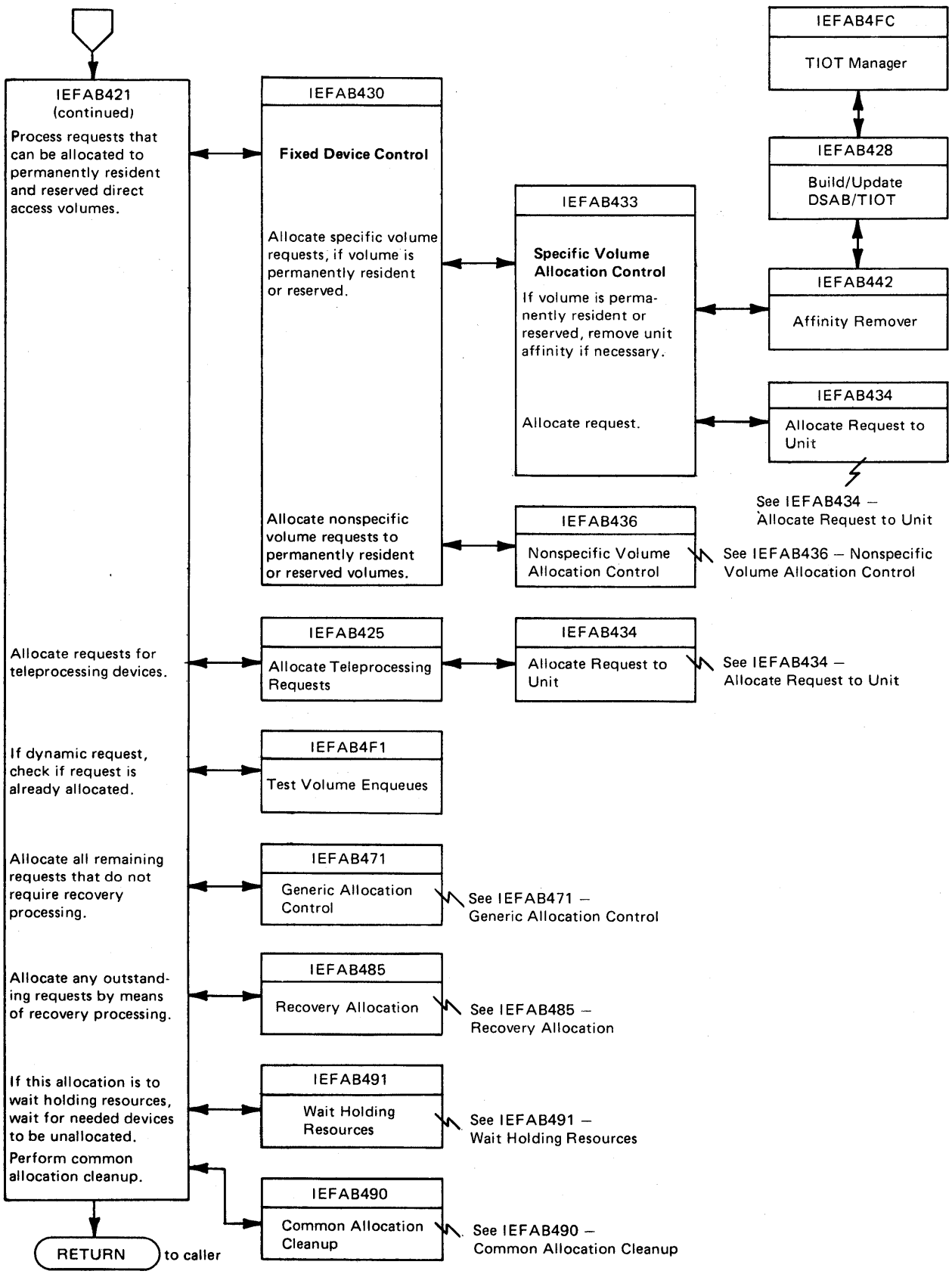
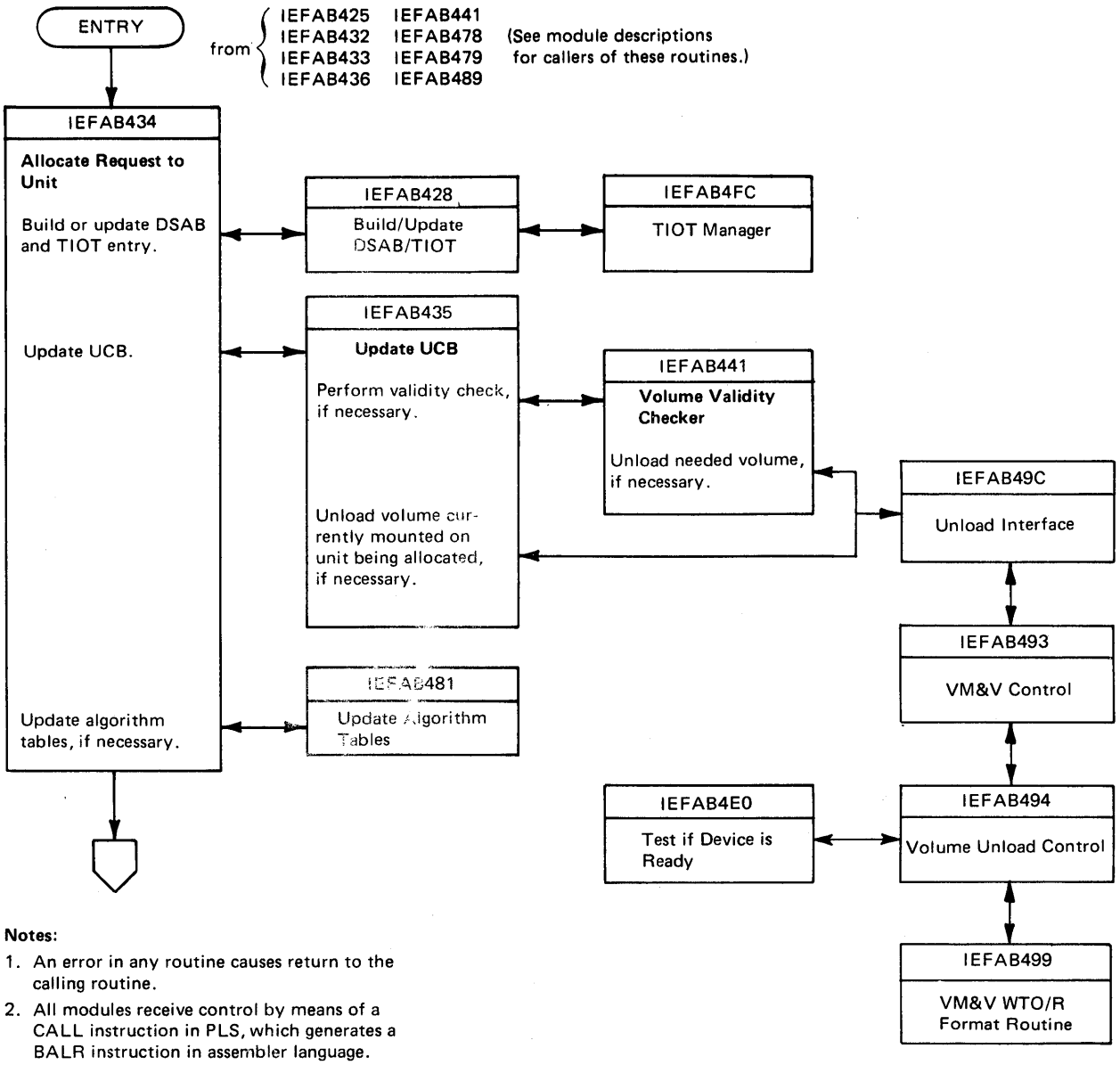


Figure 3-21. IEFAB421 – Common Allocation Control (Part 2 of 2)



Notes:

1. An error in any routine causes return to the calling routine.
2. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

Figure 3-22. IEFAB434 – Allocate Request to Unit (Part 1 of 2)

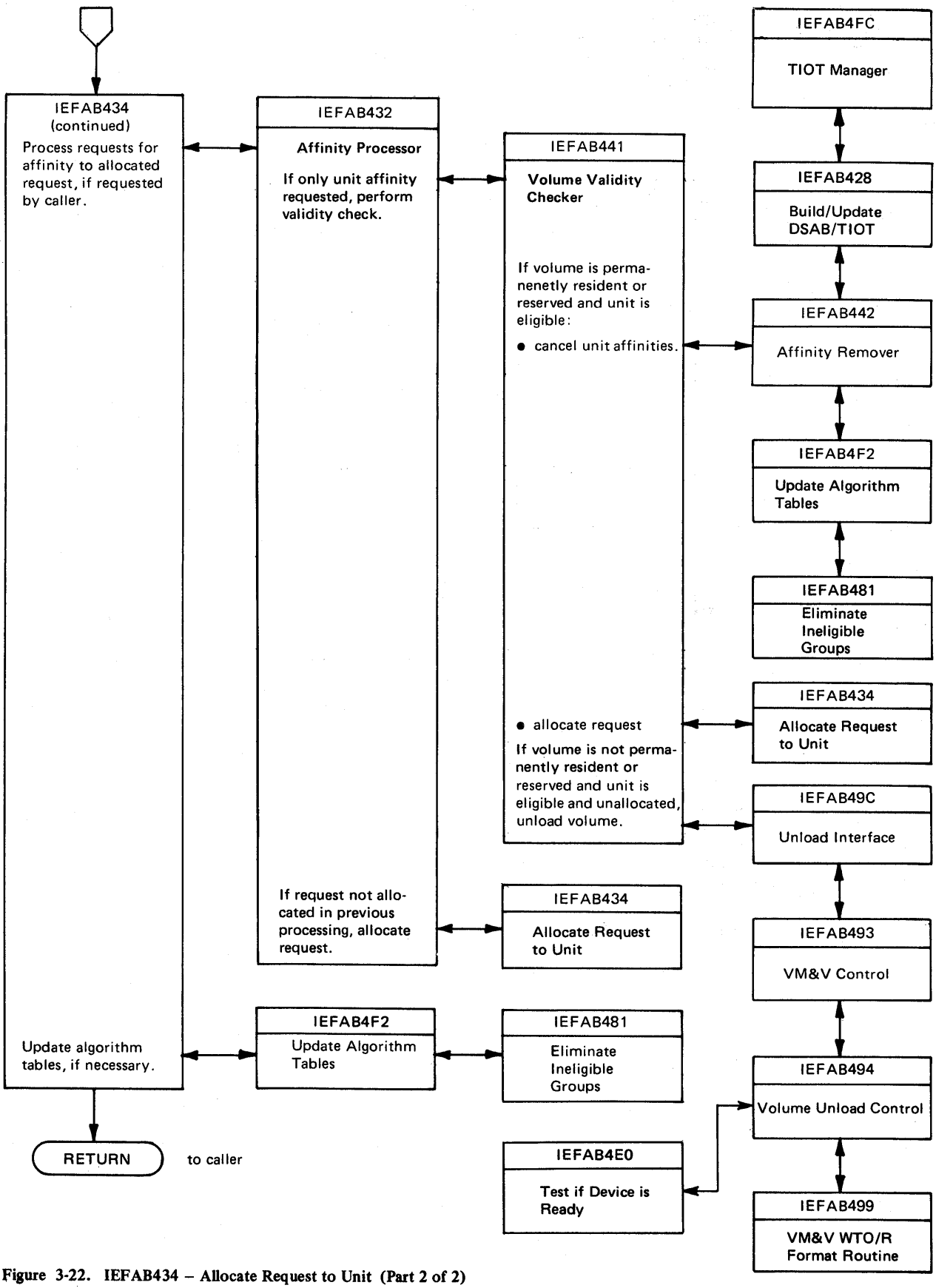
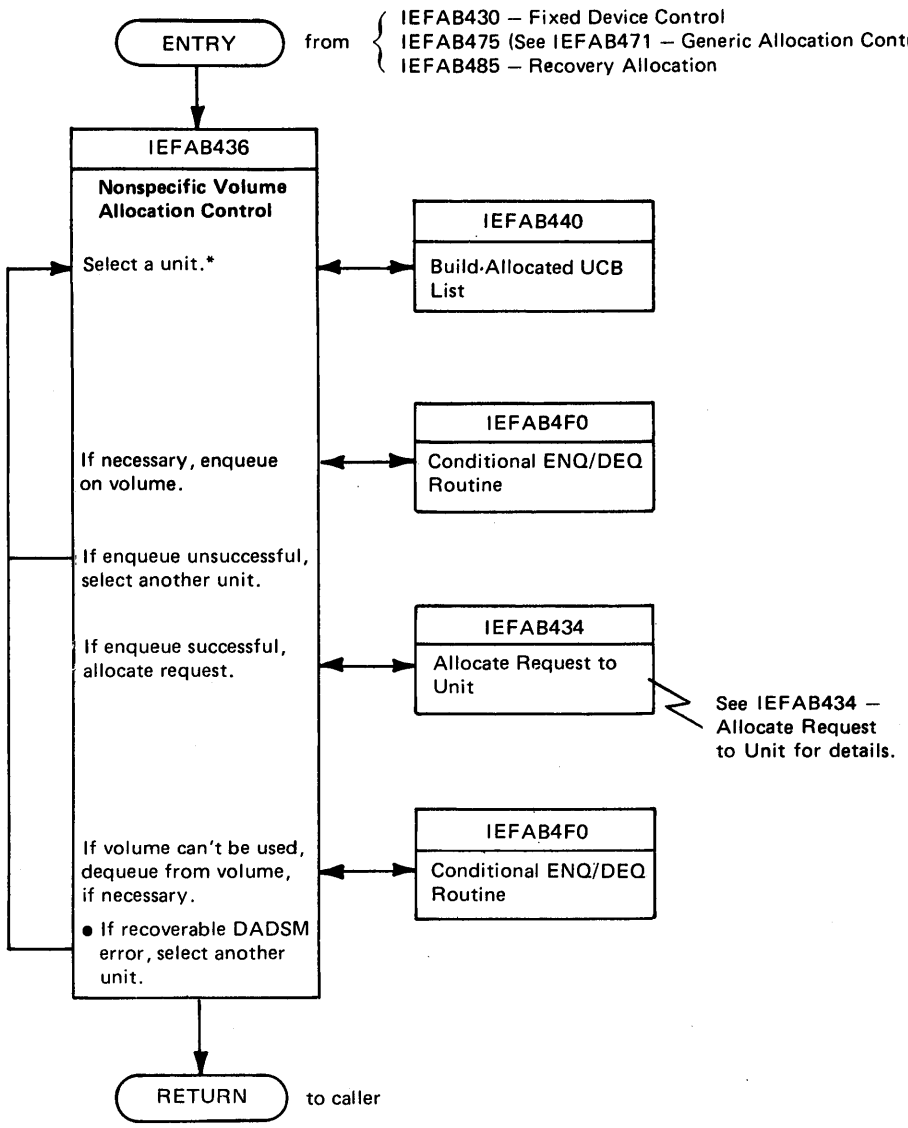


Figure 3-22. IEFAB434 – Allocate Request to Unit (Part 2 of 2)



Notes:

1. IEFAB436 processes one unit request at a time.
2. An error in any routine causes return to the calling routine.
3. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

*interfaces with System Resources Manager.

Figure 3-23. IEFAB436 – Nonspecific Volume Allocation Control

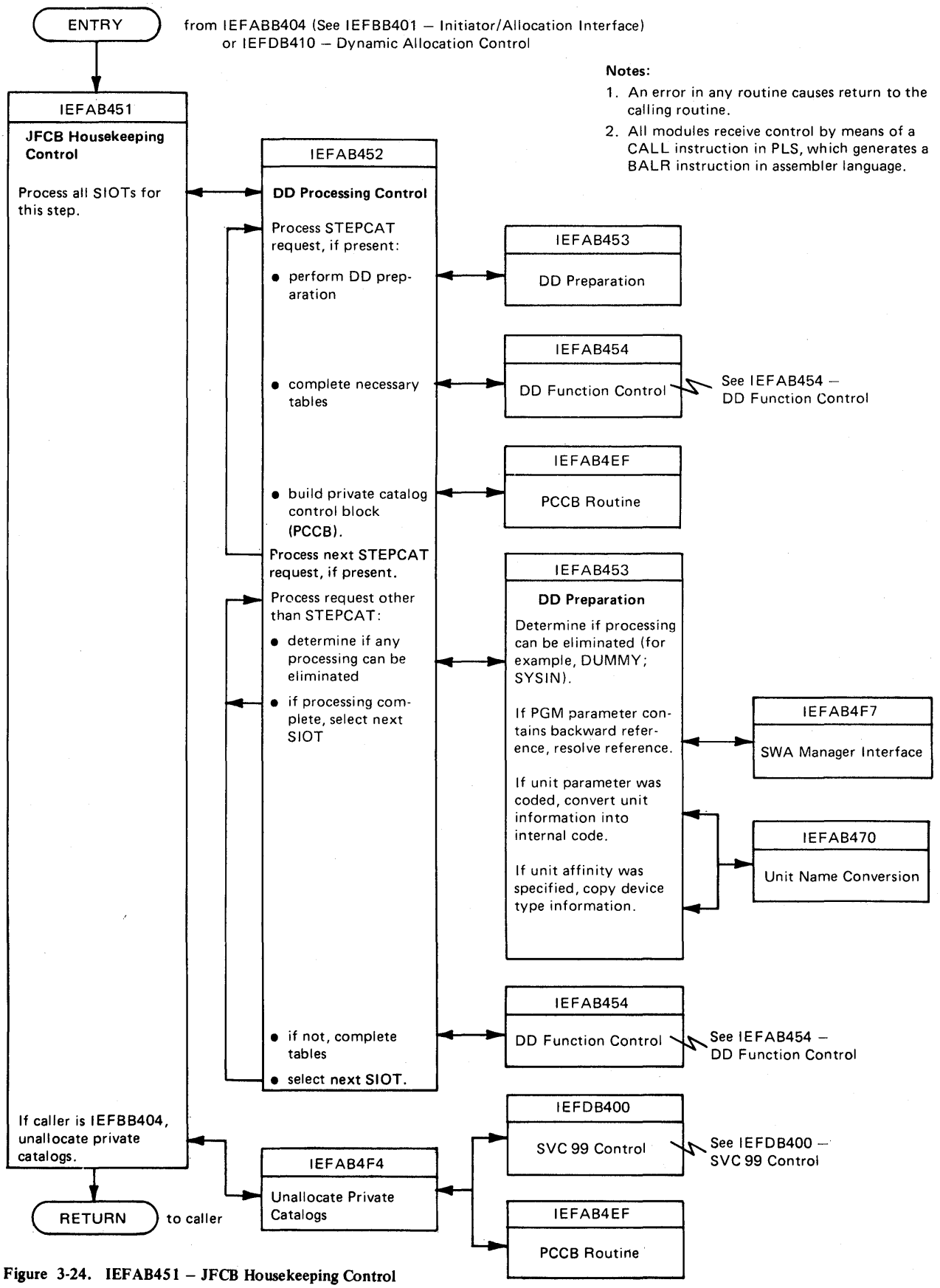
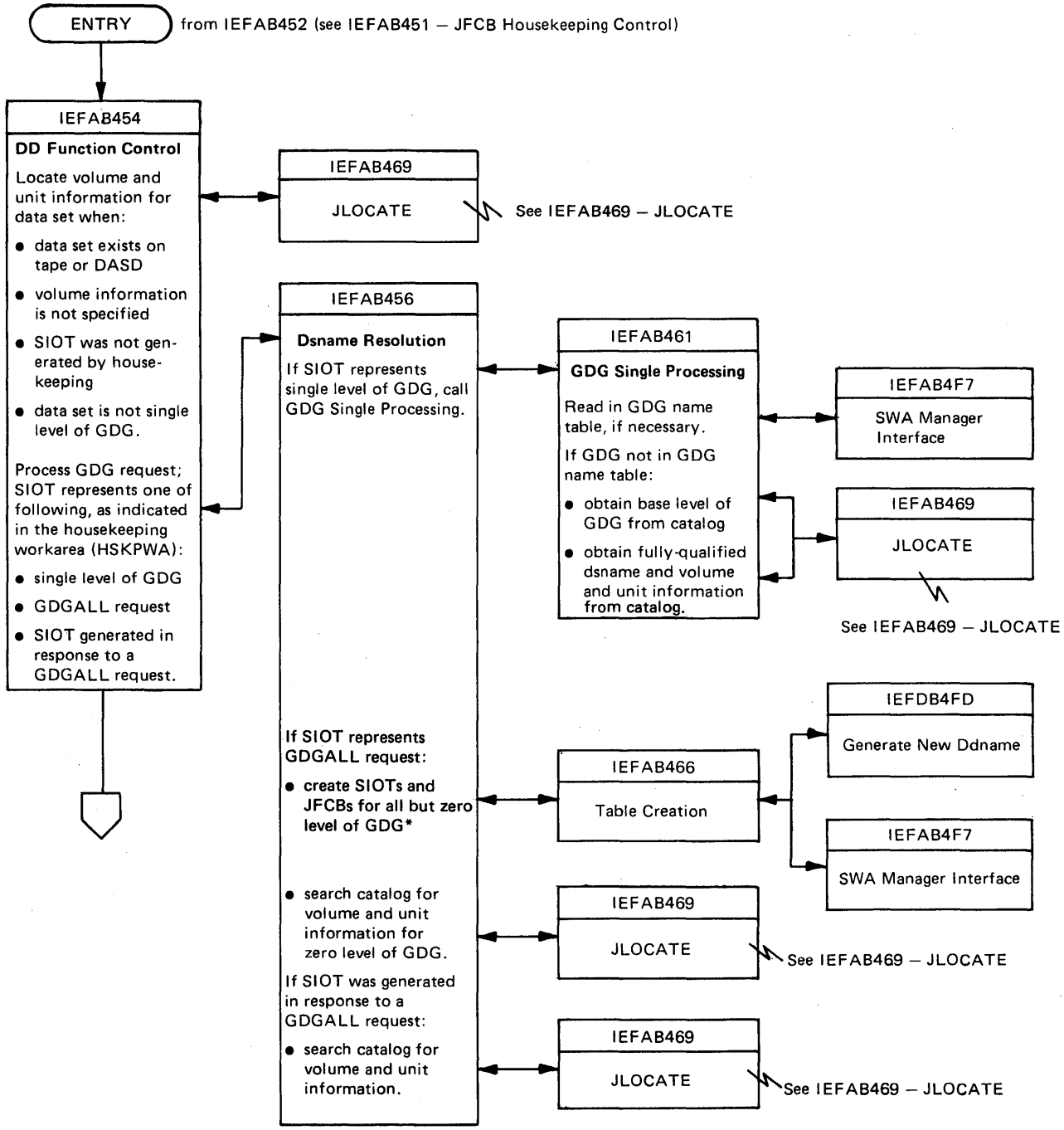
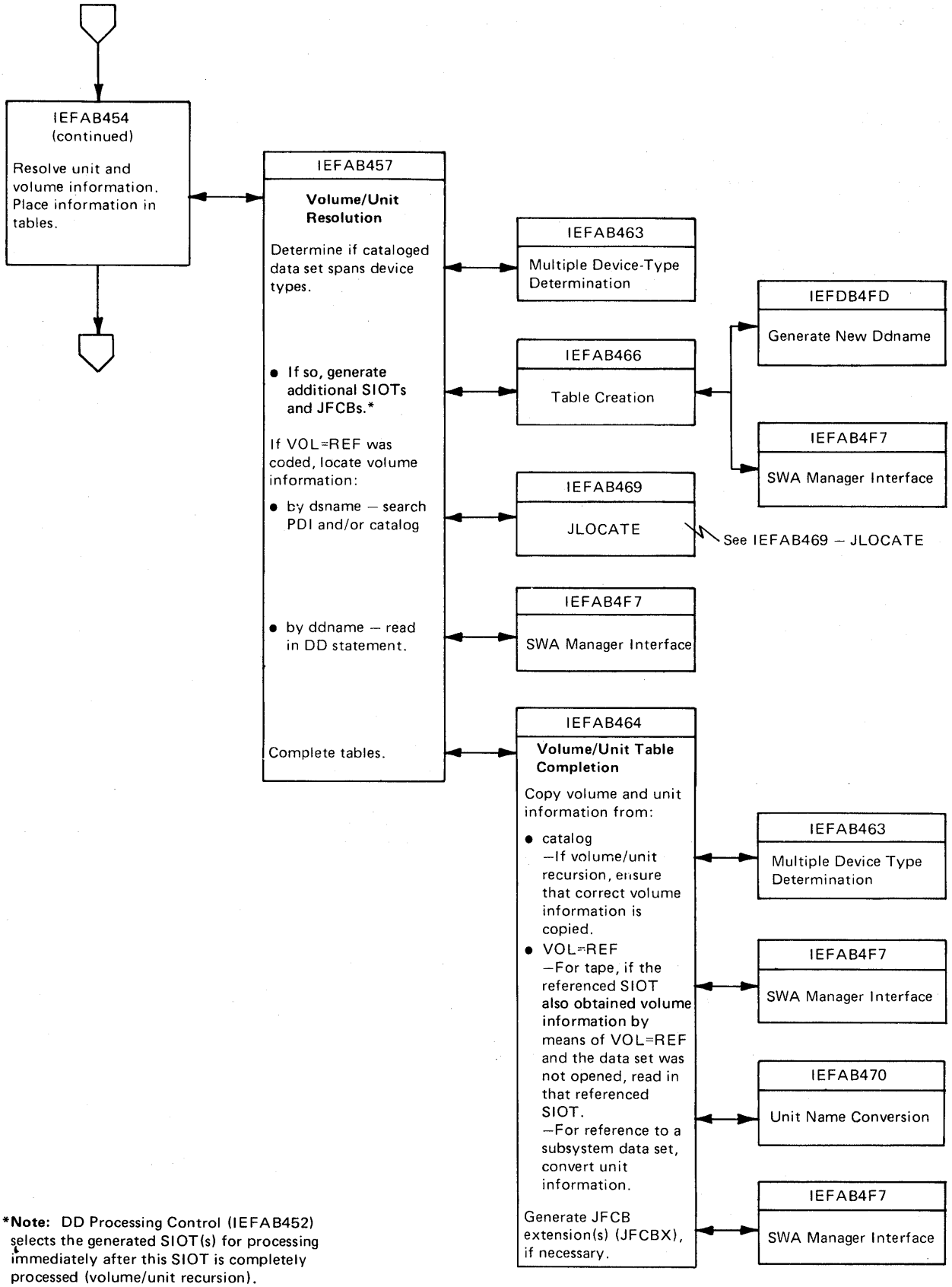


Figure 3-24. IEFAB451 – JFCB Housekeeping Control



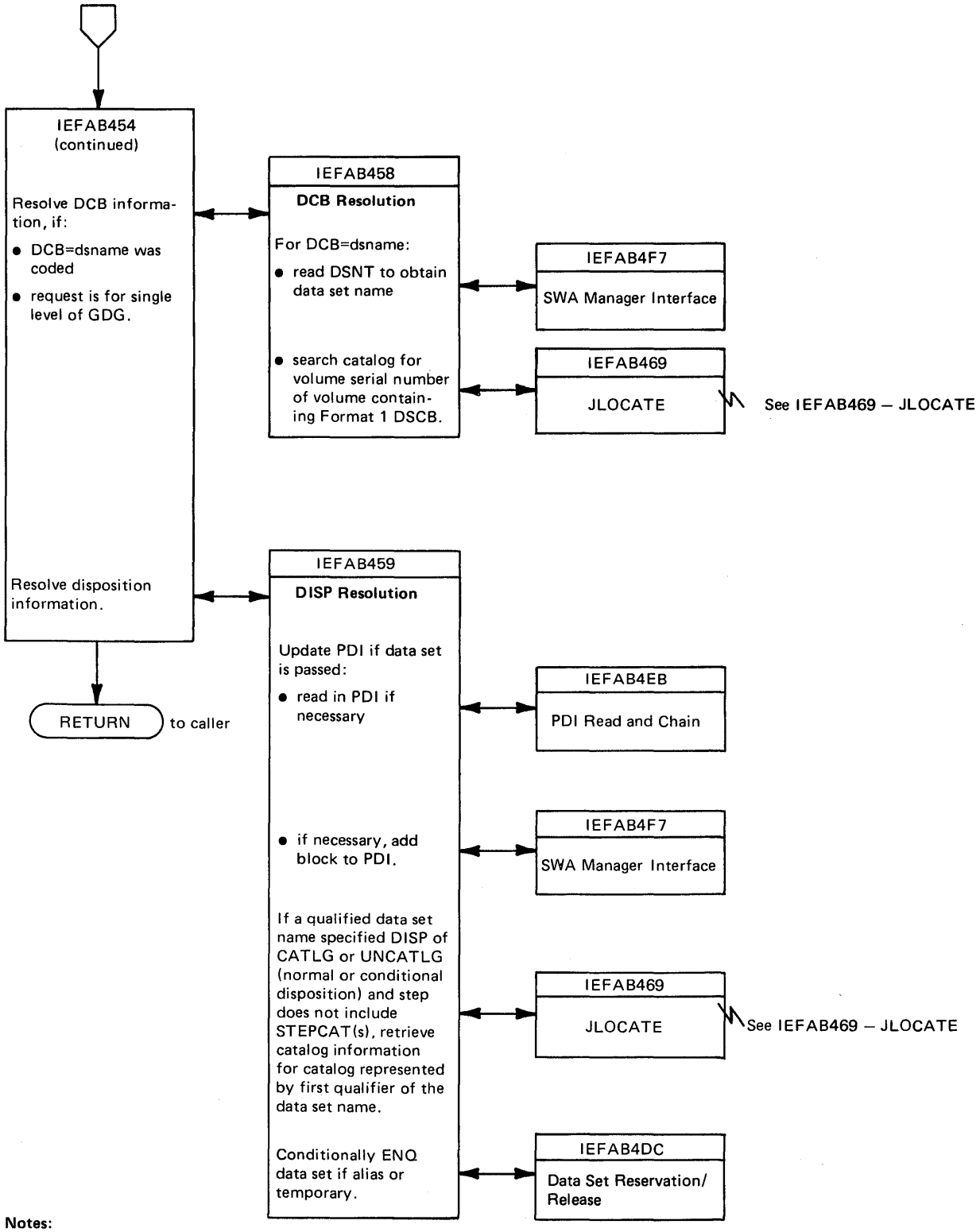
***Note:** After the zero level of the GDG is completely processed, DD Processing Control (IEFAB452) will select, one at a time, the generated SIOT(s) for the remaining level(s) (DSN recursion).

Figure 3-25. IEFAB454 – DD Function Control (Part 1 of 3)



***Note:** DD Processing Control (IEFAB452) selects the generated SIOT(s) for processing immediately after this SIOT is completely processed (volume/unit recursion).

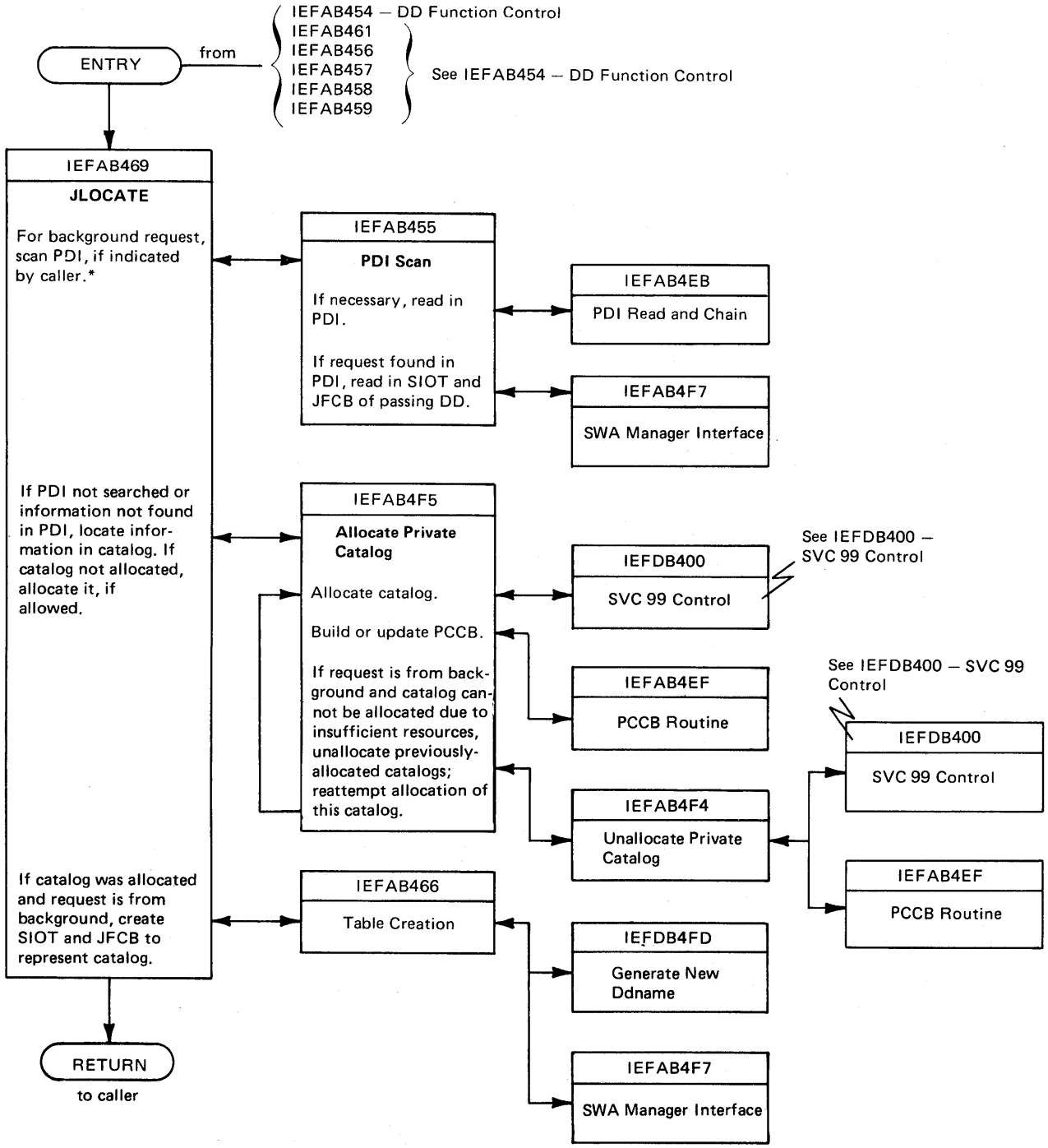
Figure 3-25. IEFAB454 – DD Function Control (Part 2 of 3)



Notes:

1. An error in any routine causes return to the calling routine.
2. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

Figure 3-25. IEFAB454 – DD Function Control (Part 3 of 3)

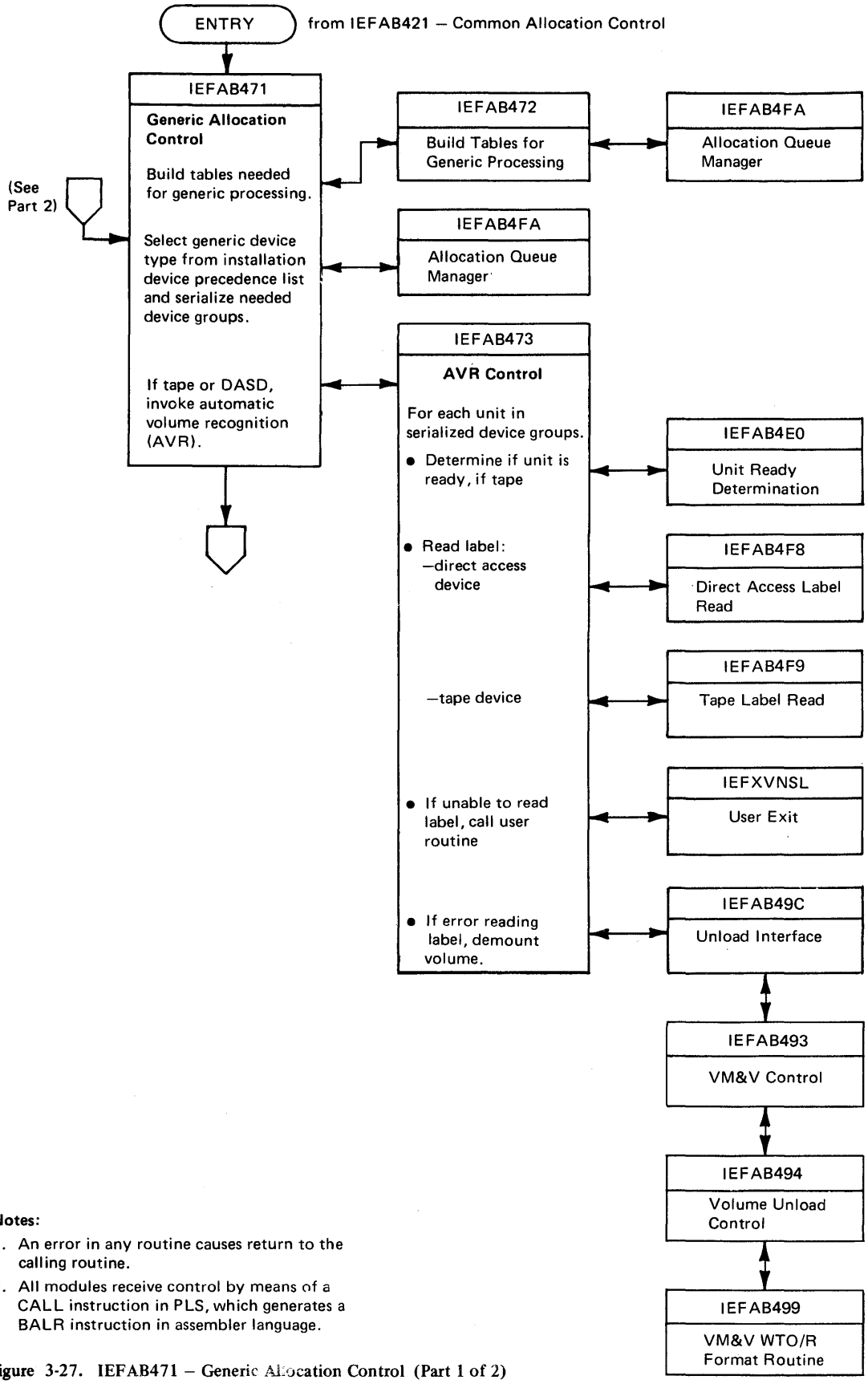


Notes:

1. An error in any routine causes return to the calling routine.
2. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

*PDI scan can be indicated only by IEFAB454, IEFAB457, or IEFAB461.

Figure 3-26. IEFAB469 – JLOCATE



Notes:

1. An error in any routine causes return to the calling routine.
2. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

Figure 3-27. IEFAB471 – Generic Allocation Control (Part 1 of 2)

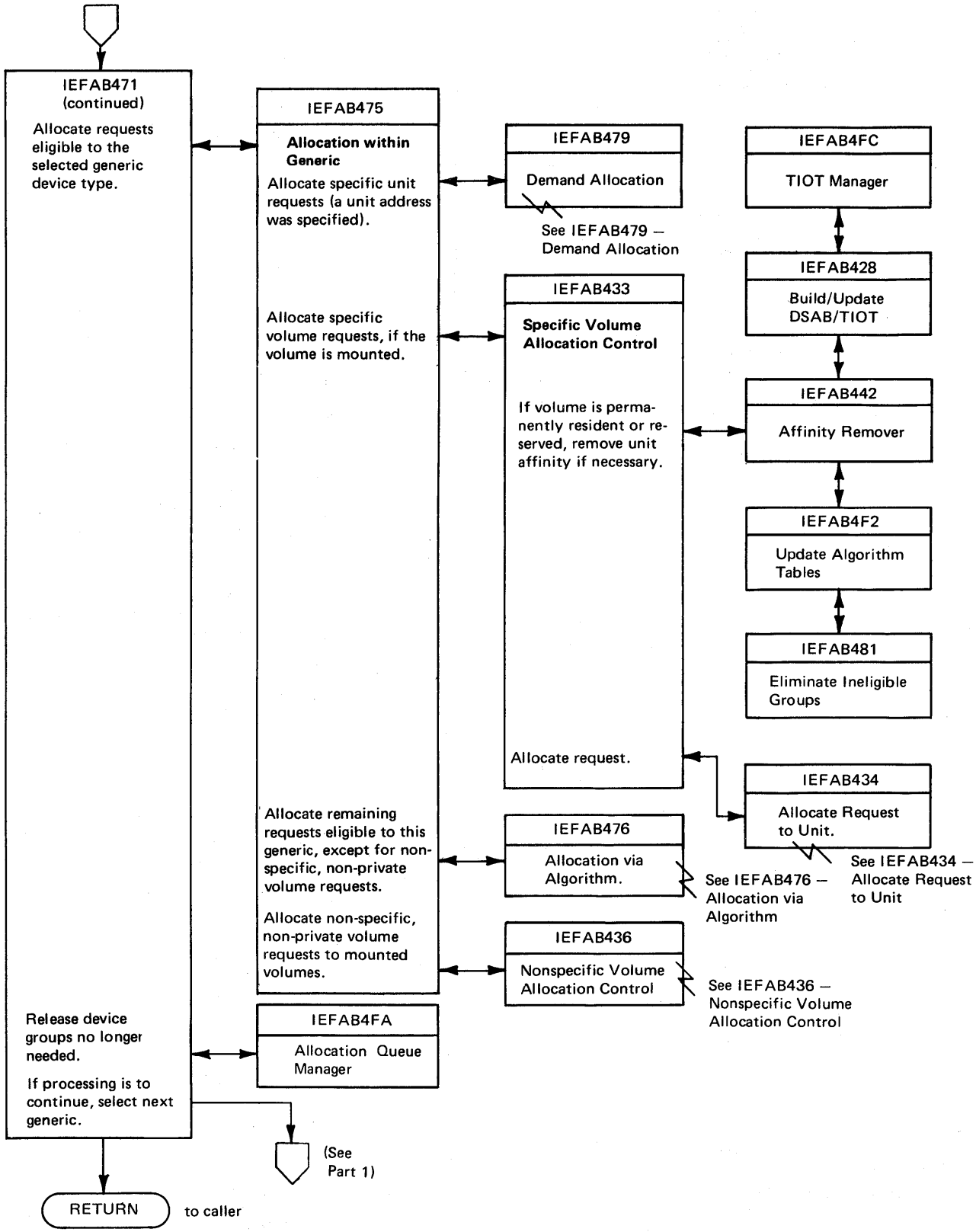
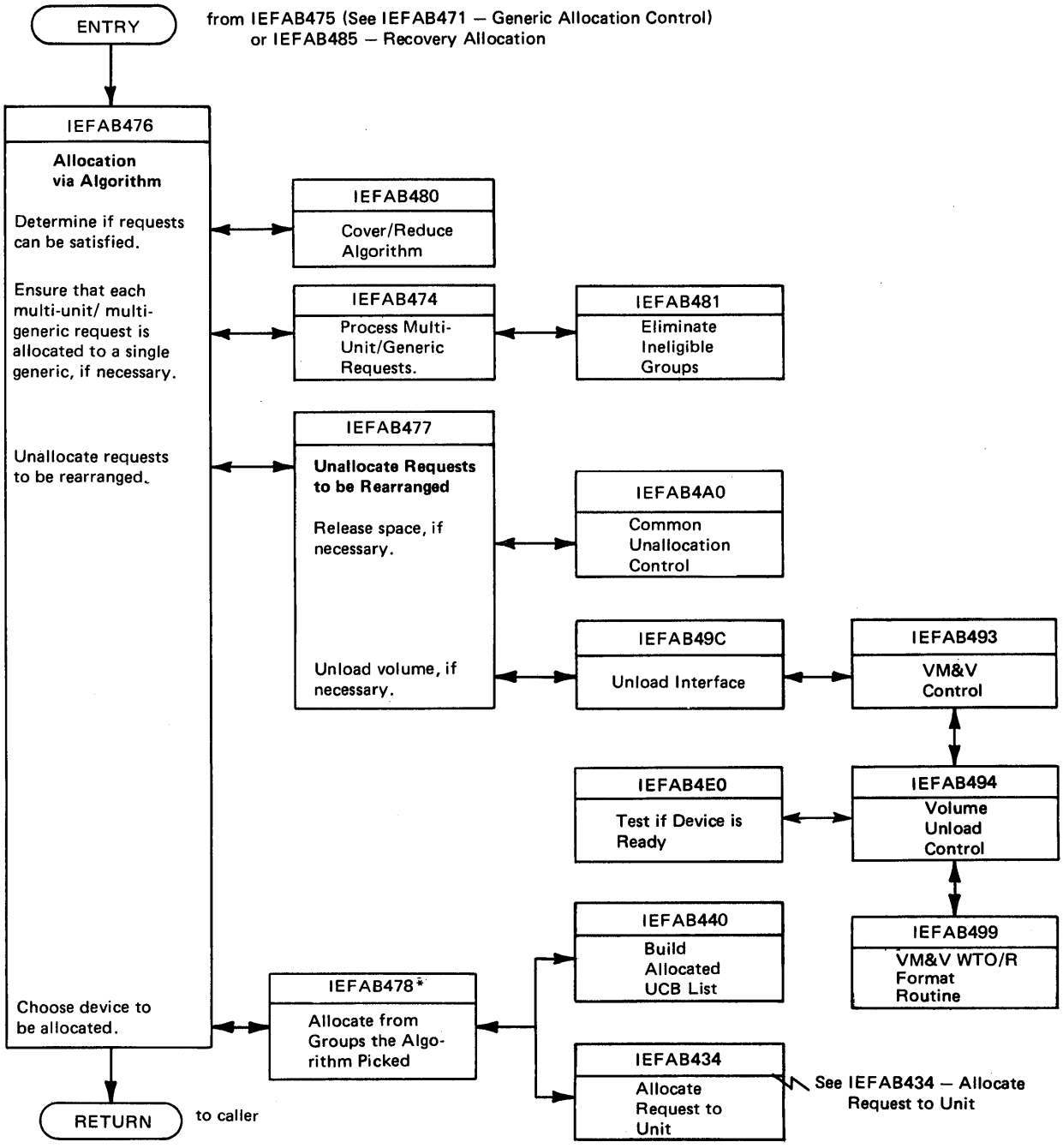


Figure 3-27. IEFAB471 – Generic Allocation Control (Part 2 of 2)



Notes:

1. An error in any routine causes return to the calling routine.
2. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

*checks with System Resources Manager and MSS Mount Equalization

Figure 3-28. IEFAB476 – Allocation via Algorithm

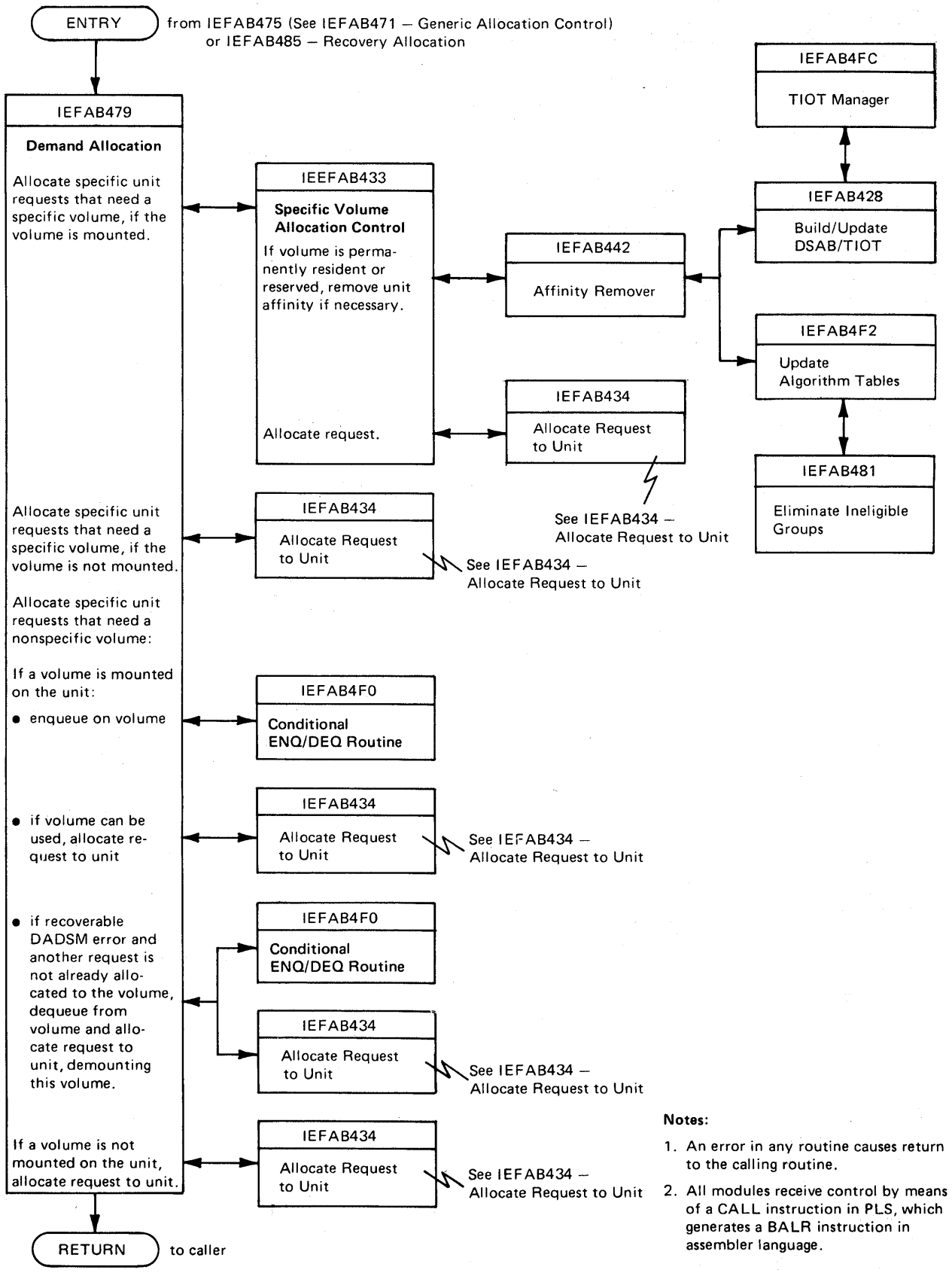


Figure 3-29. IEFAB479 – Demand Allocation

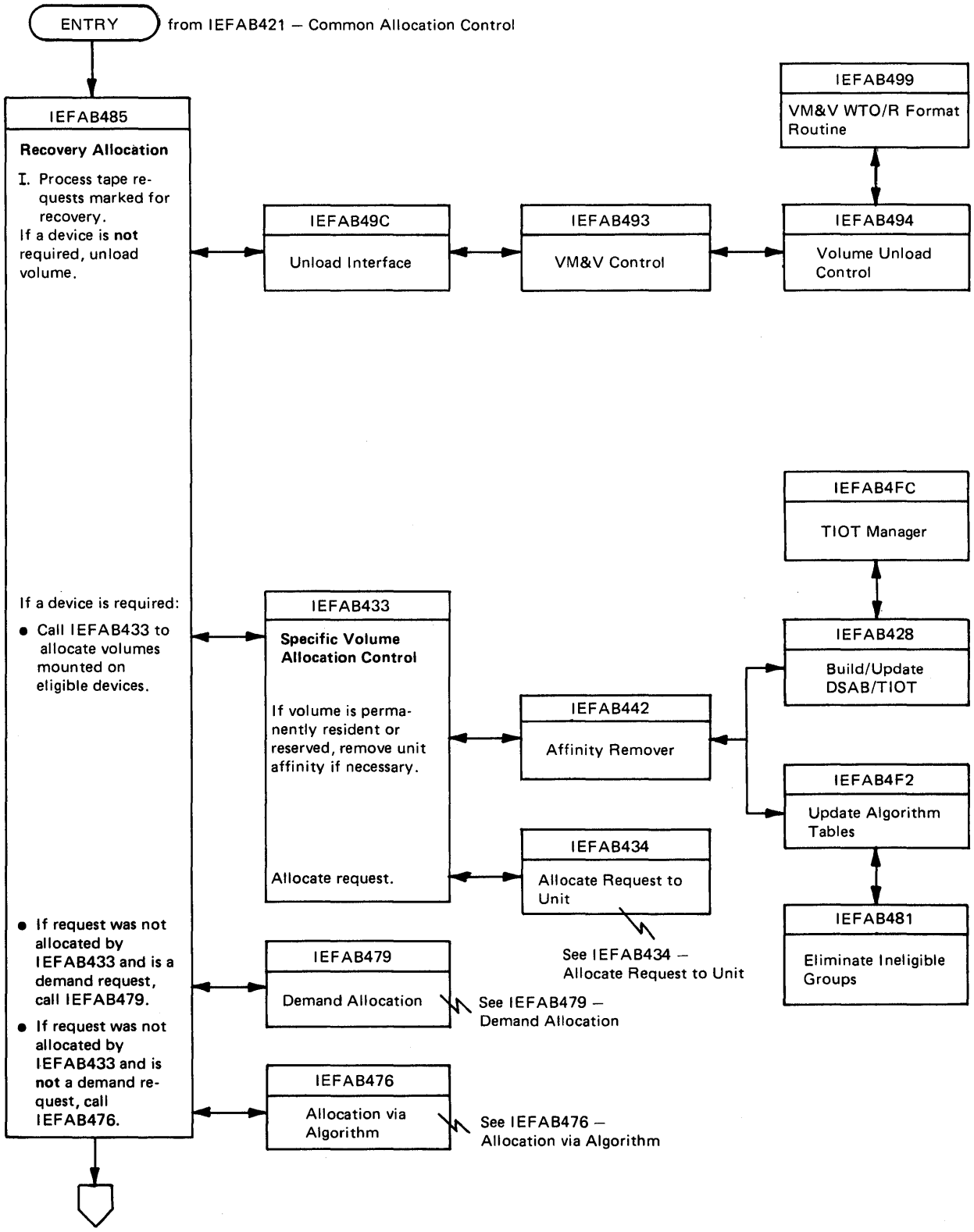
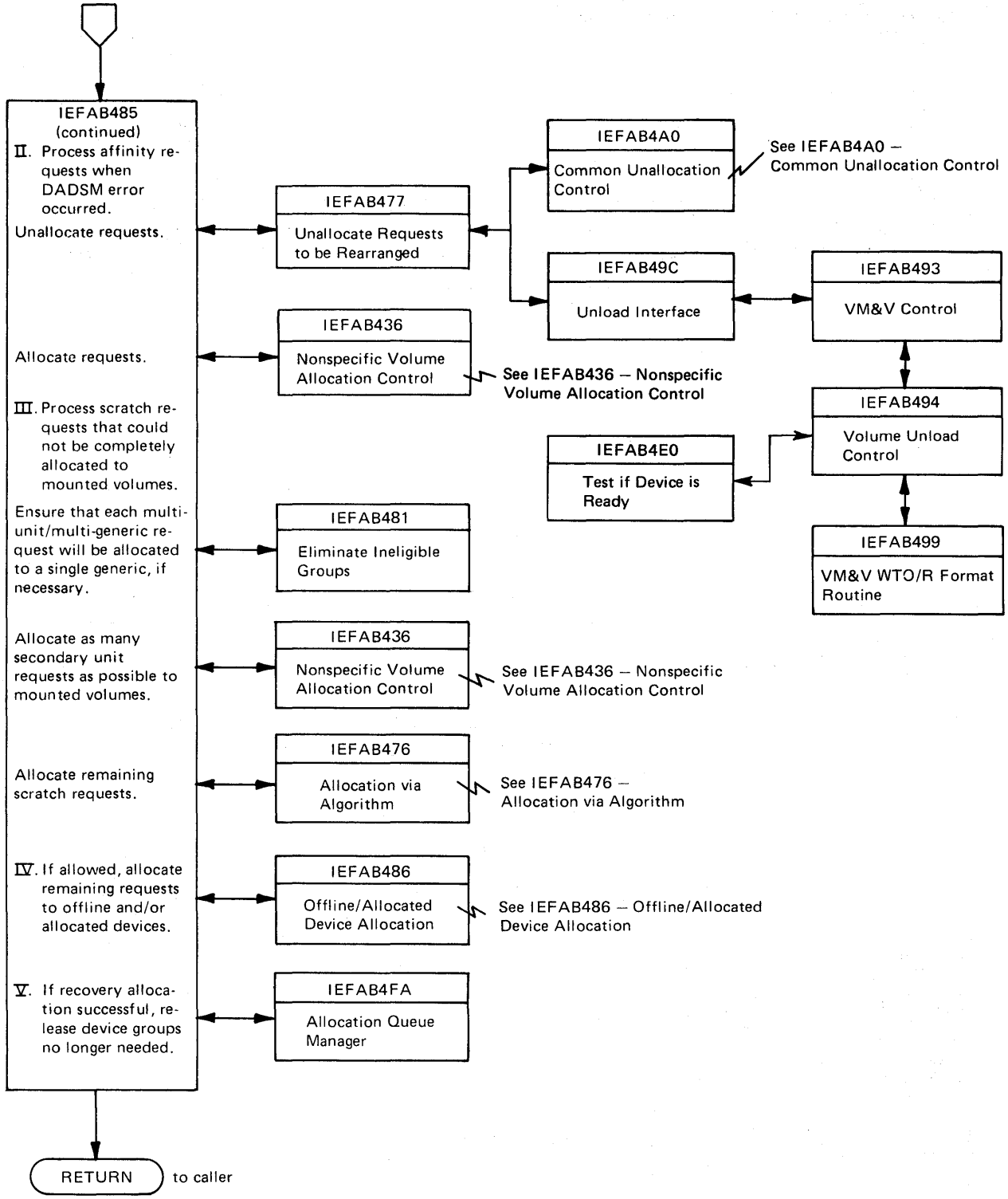


Figure 3-30. IEFAB485 – Recovery Allocation (Part 1 of 2)



Notes:

1. An error in any routine causes return to the calling routine.
2. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

Figure 3-30. IEFAB485 – Recovery Allocation (Part 2 of 2)

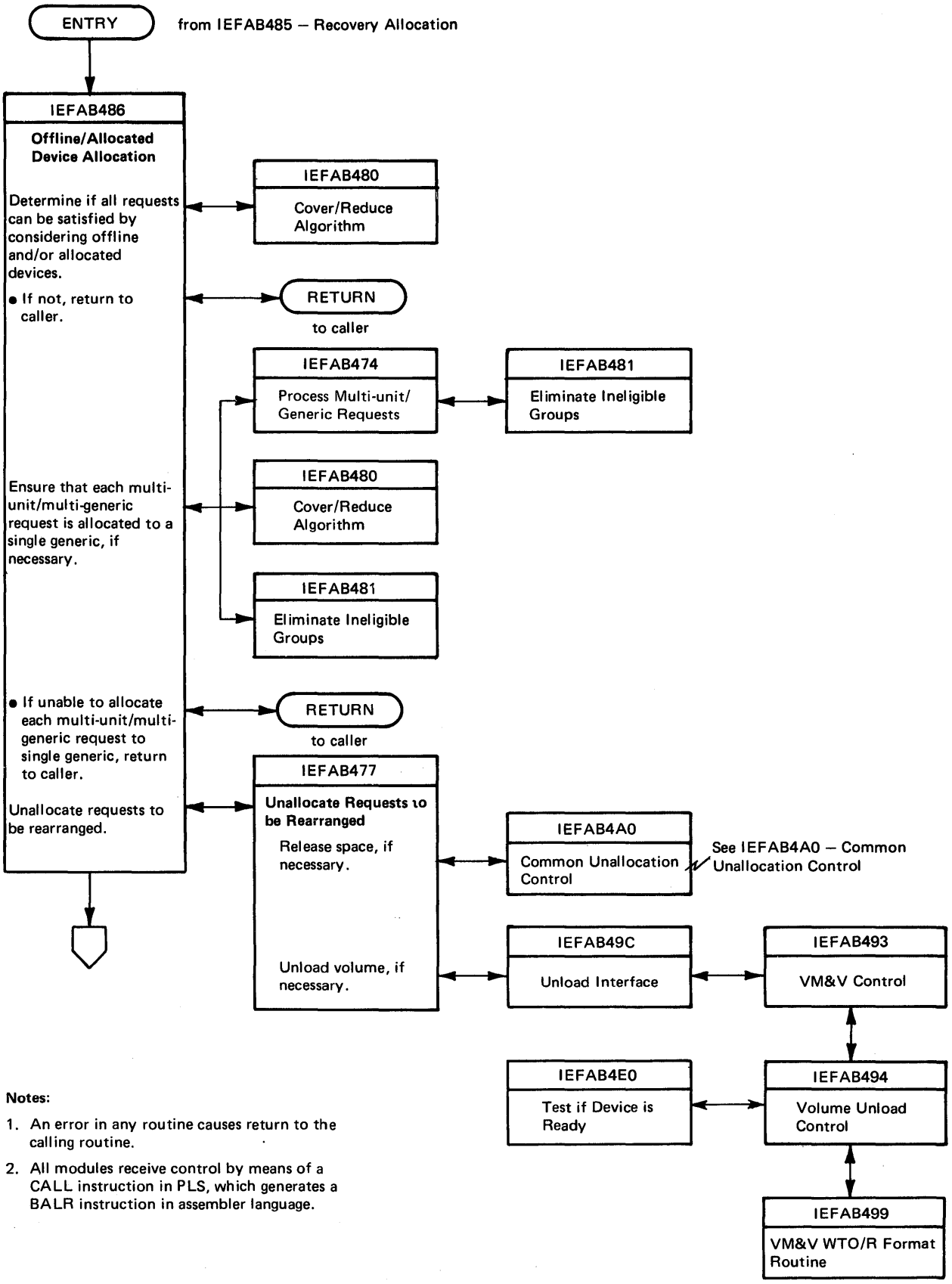


Figure 3-31. IEFAB486 – Offline/Allocated Device Allocation (Part 1 of 4)

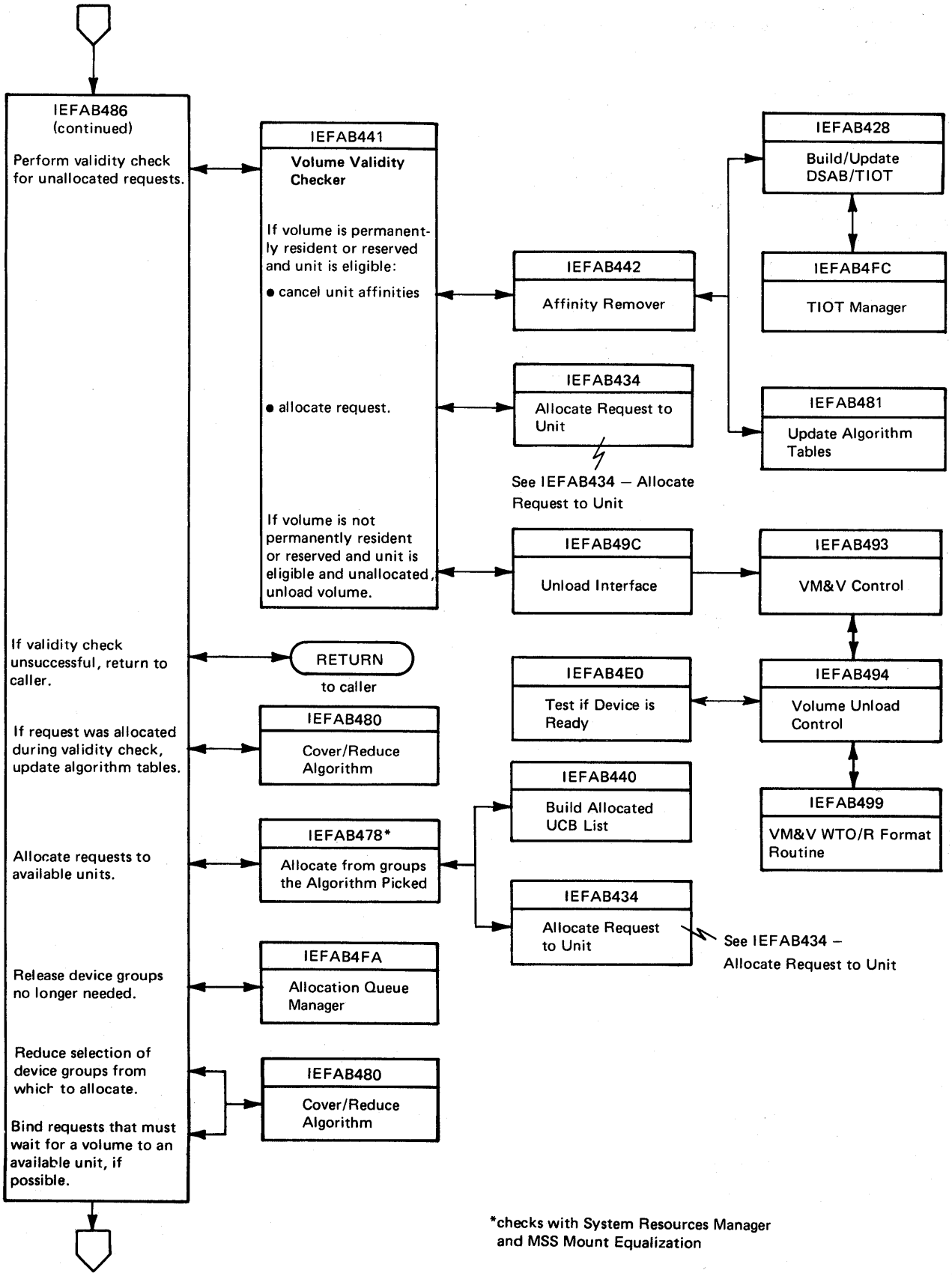


Figure 3-31. IEFAB486 – Offline/Allocated Device Allocation (Part 2 of 4)

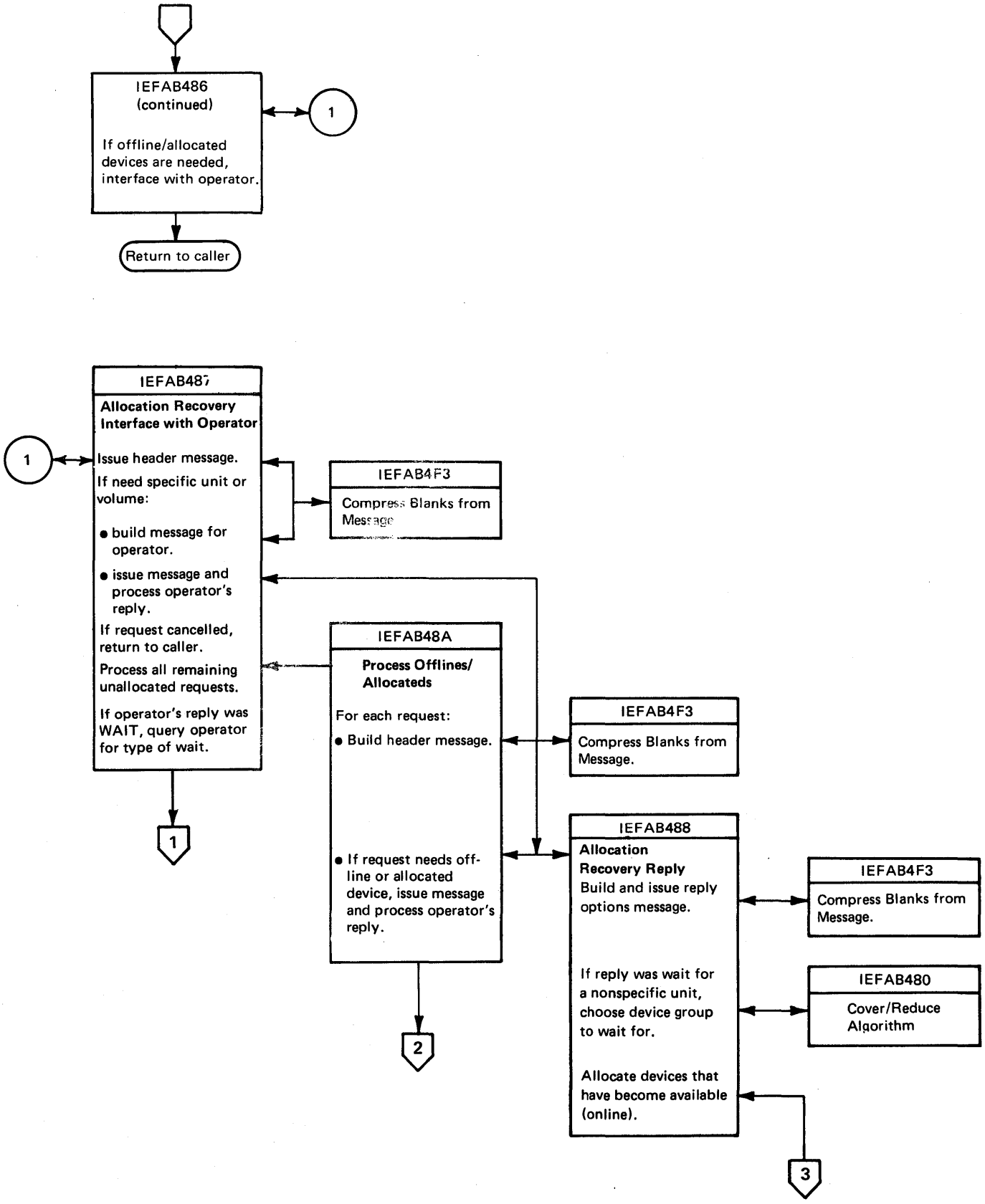
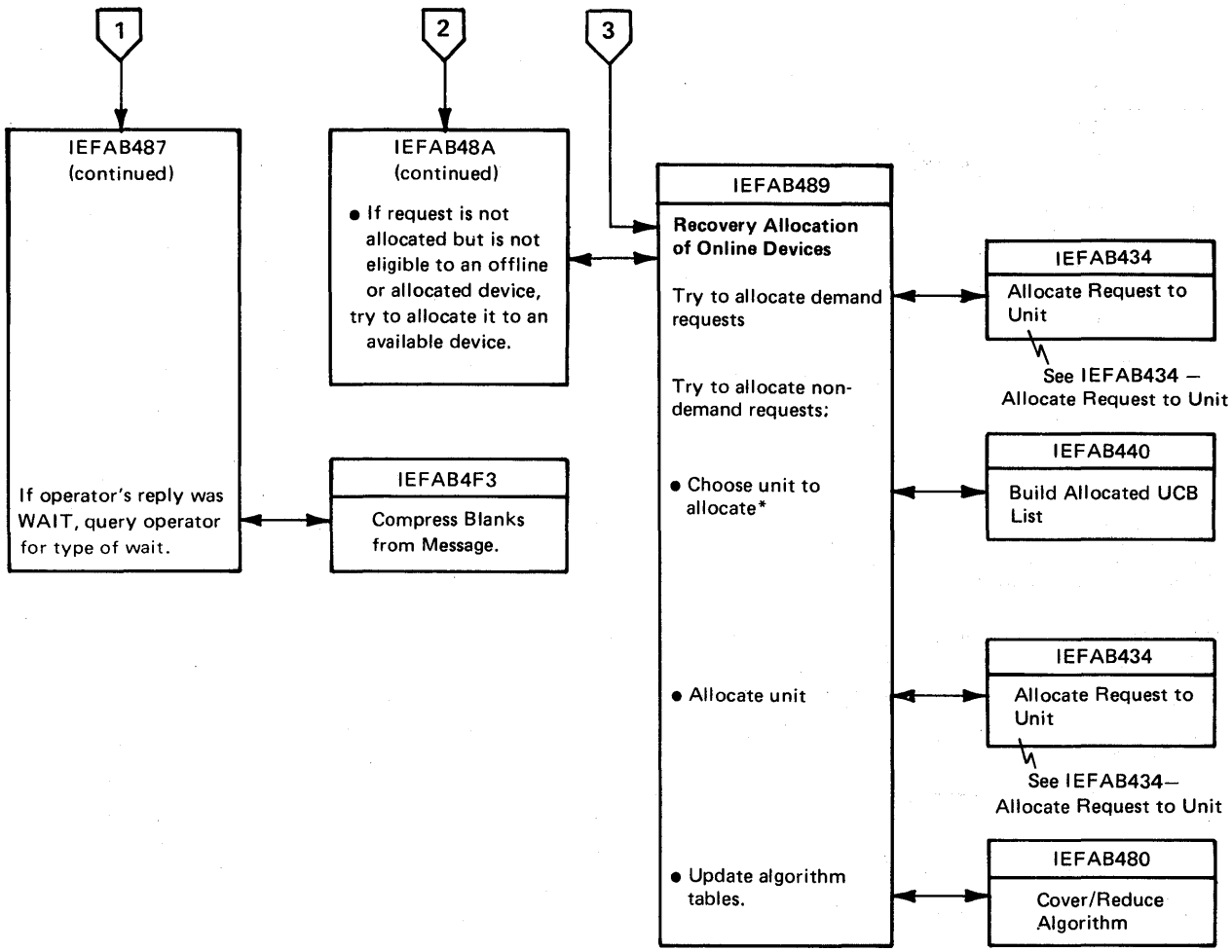


Figure 3-31. IEFAB486 – Offline/Allocated Device Allocation (Part 3 of 4)



*checks with System Resources Manager and MSS Mount Equalization

Figure 3-31. IEFAB486 – Offline/Allocated Device Allocation (Part 4 of 4)

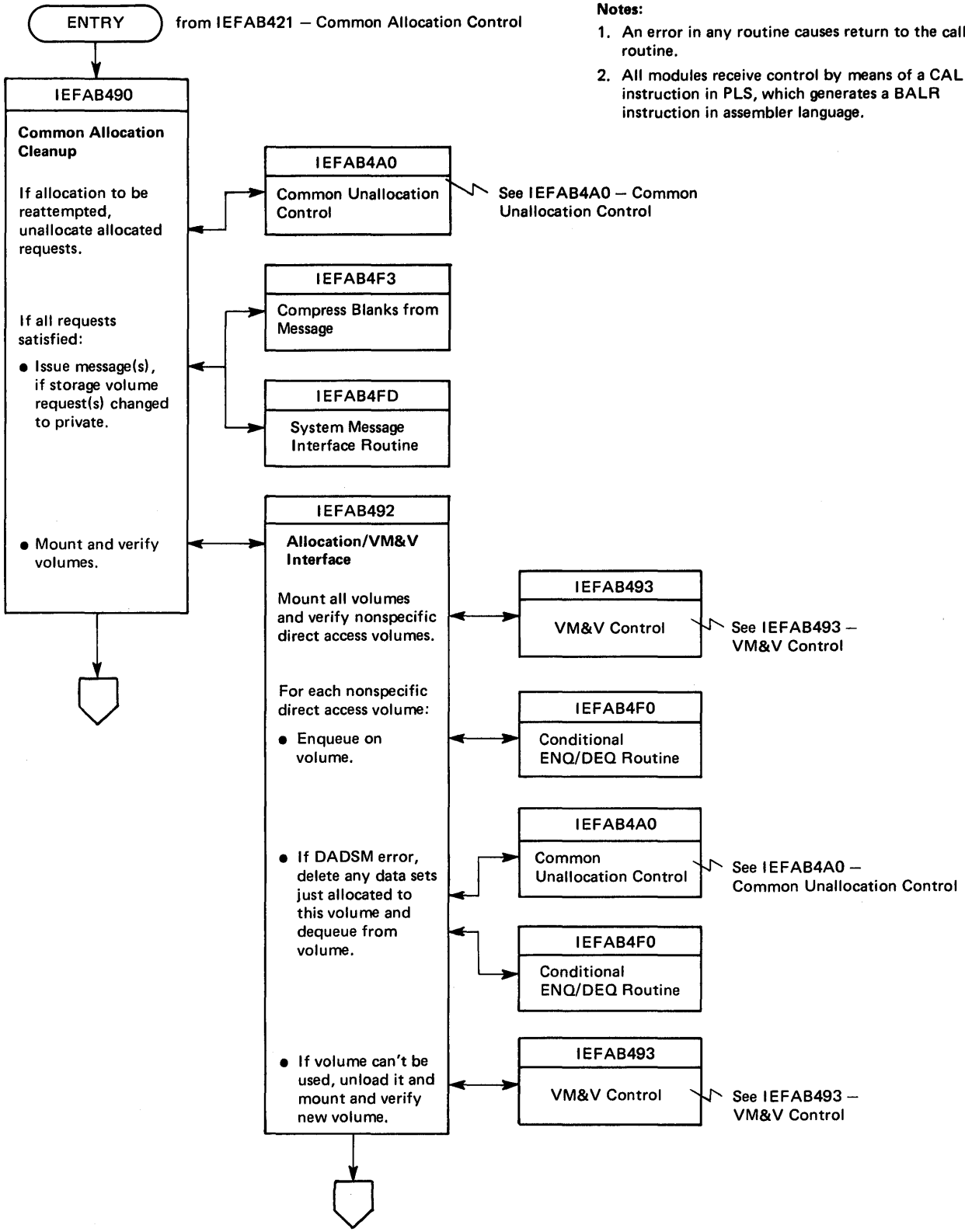


Figure 3-32. IEFAB490 – Common Allocation Cleanup (Part 1 of 2)

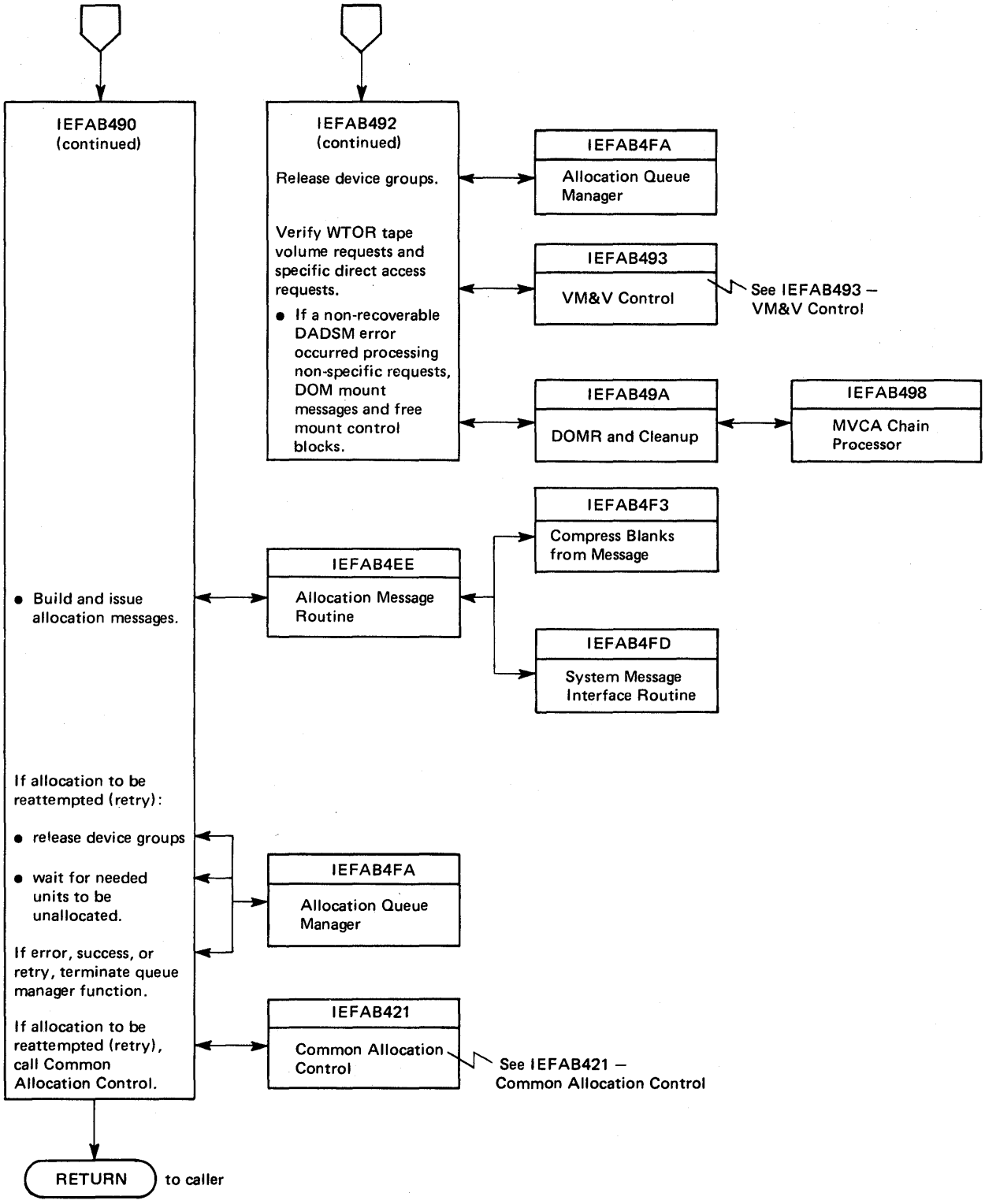


Figure 3-32. IEFAB490 – Common Allocation Cleanup (Part 2 of 2)

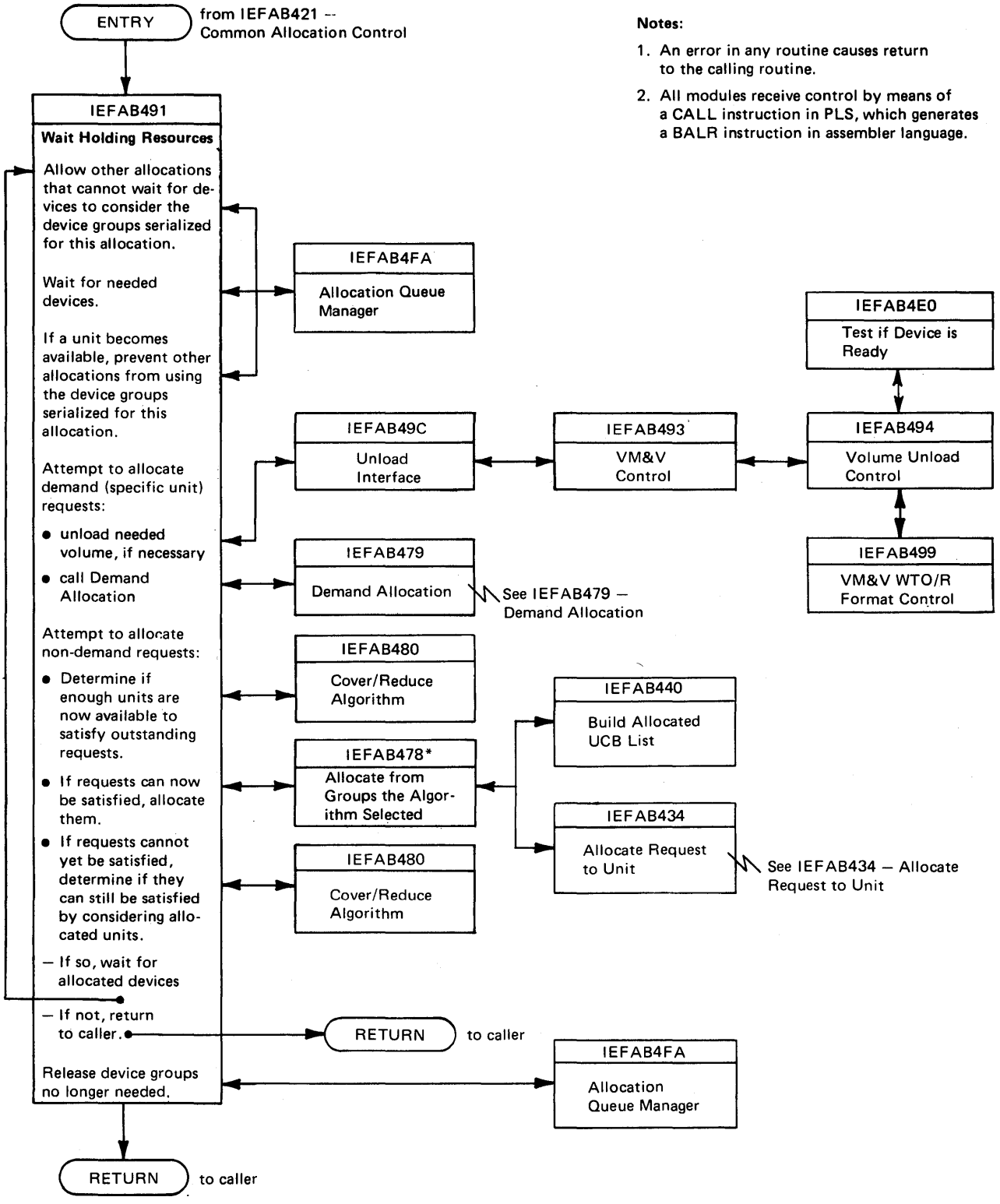


Figure 3-33. IEFAB491 – Wait Holding Resources

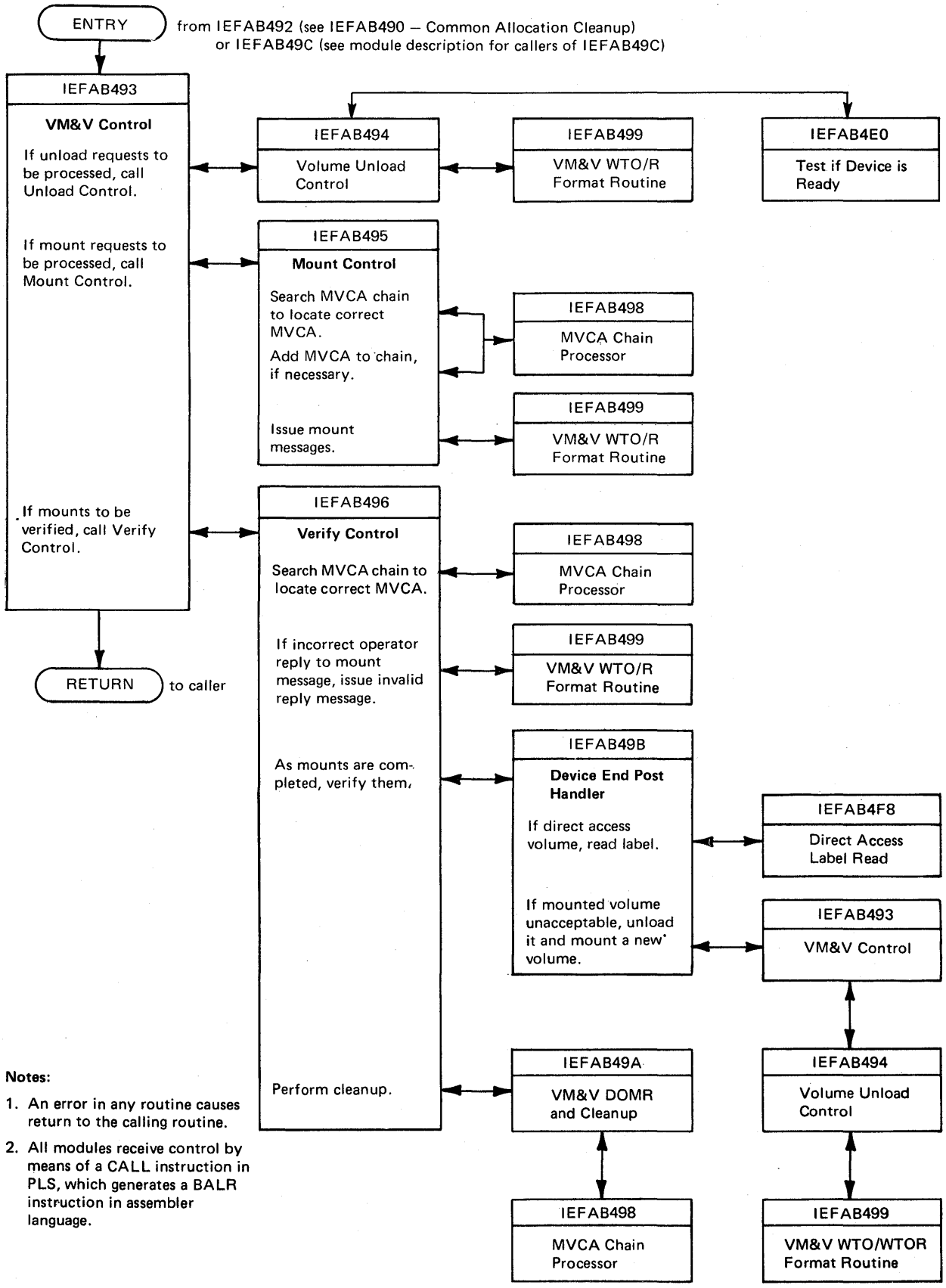


Figure 3-34. IEFAB493 – VM&V (Volume Mount and Verify) Control

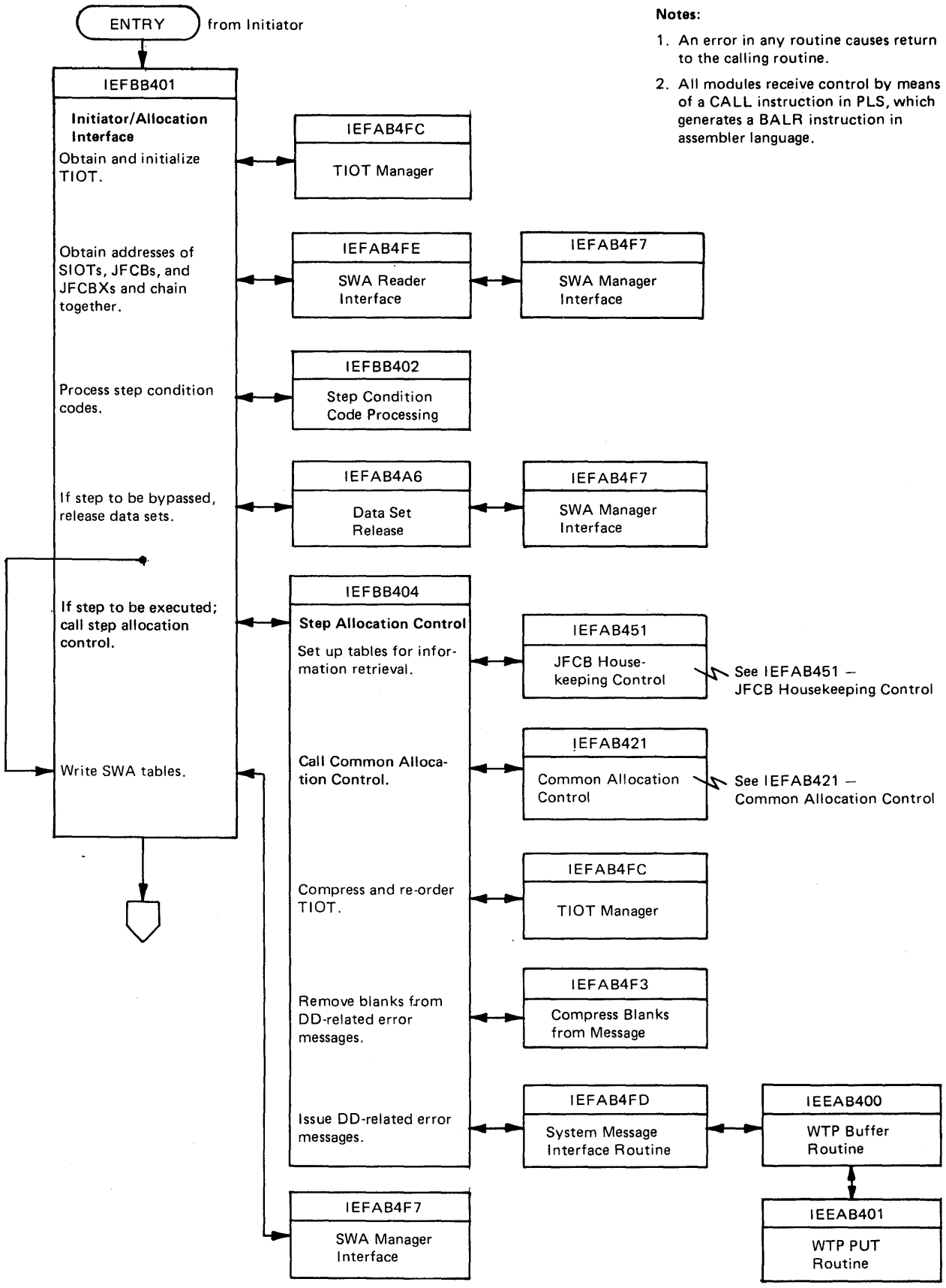


Figure 3-35. IEFBB401 – Initiator/Allocation Interface (Part 1 of 2)

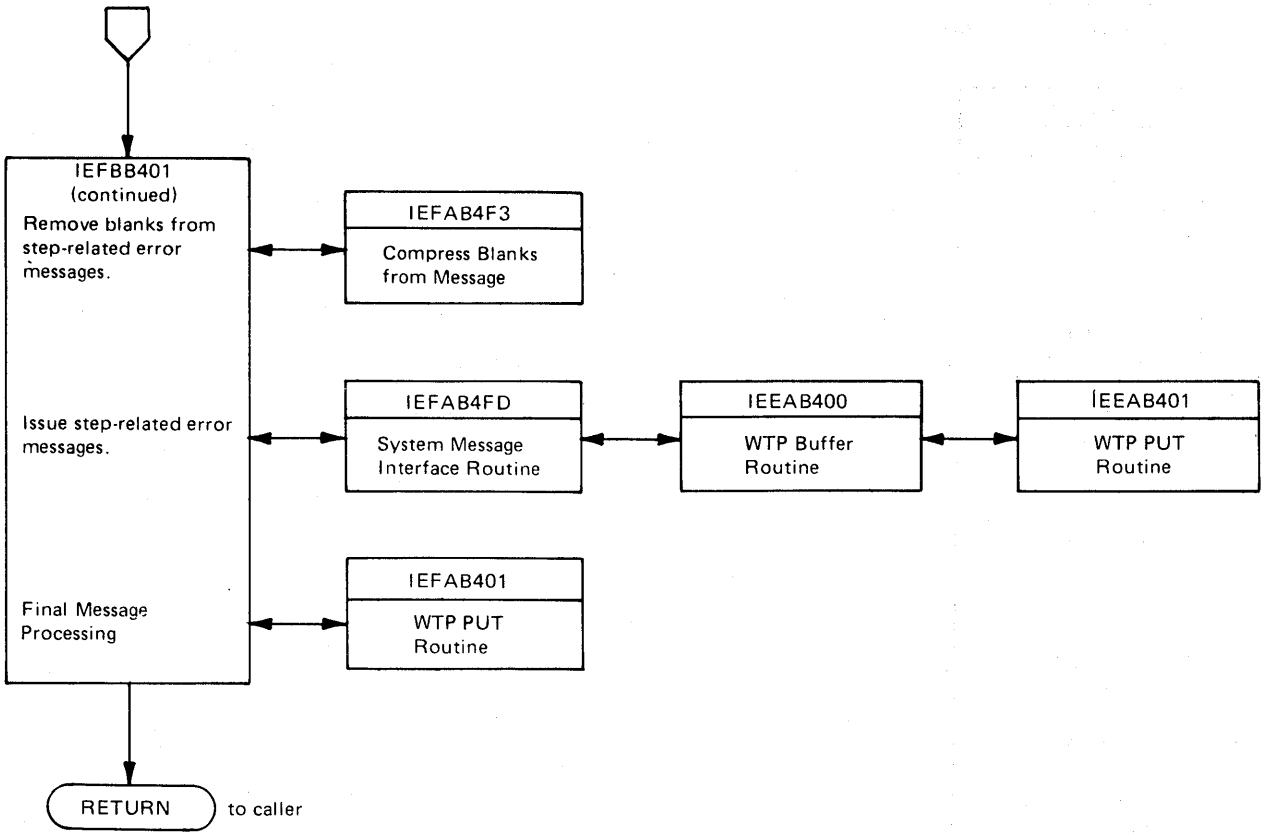


Figure 3-35. IEFBB401 – Initiator/Allocation Interface (Part 2 of 2)

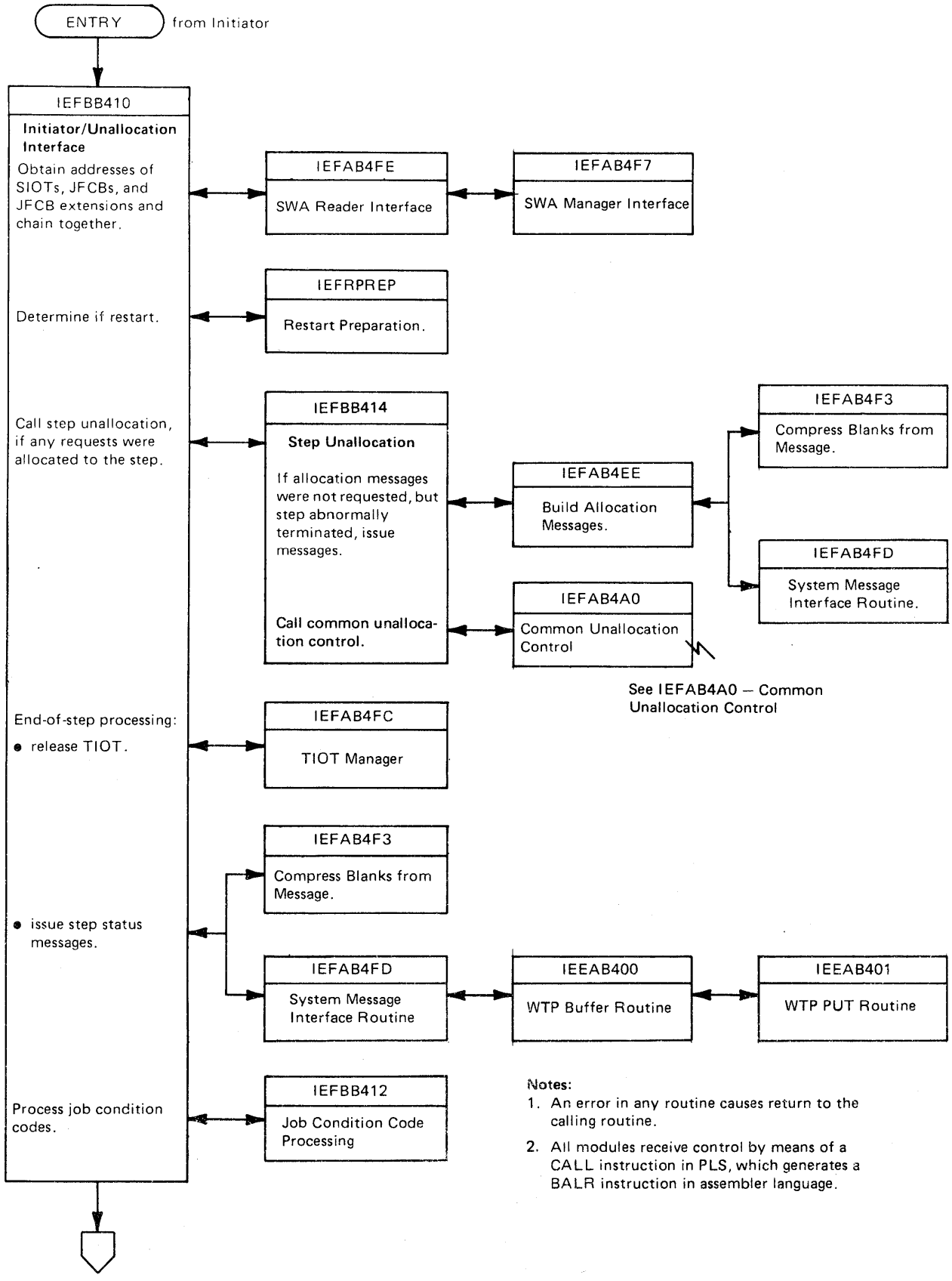


Figure 3-36. IEFBB410 – Initiator/Unallocation Interface (Part 1 of 3)

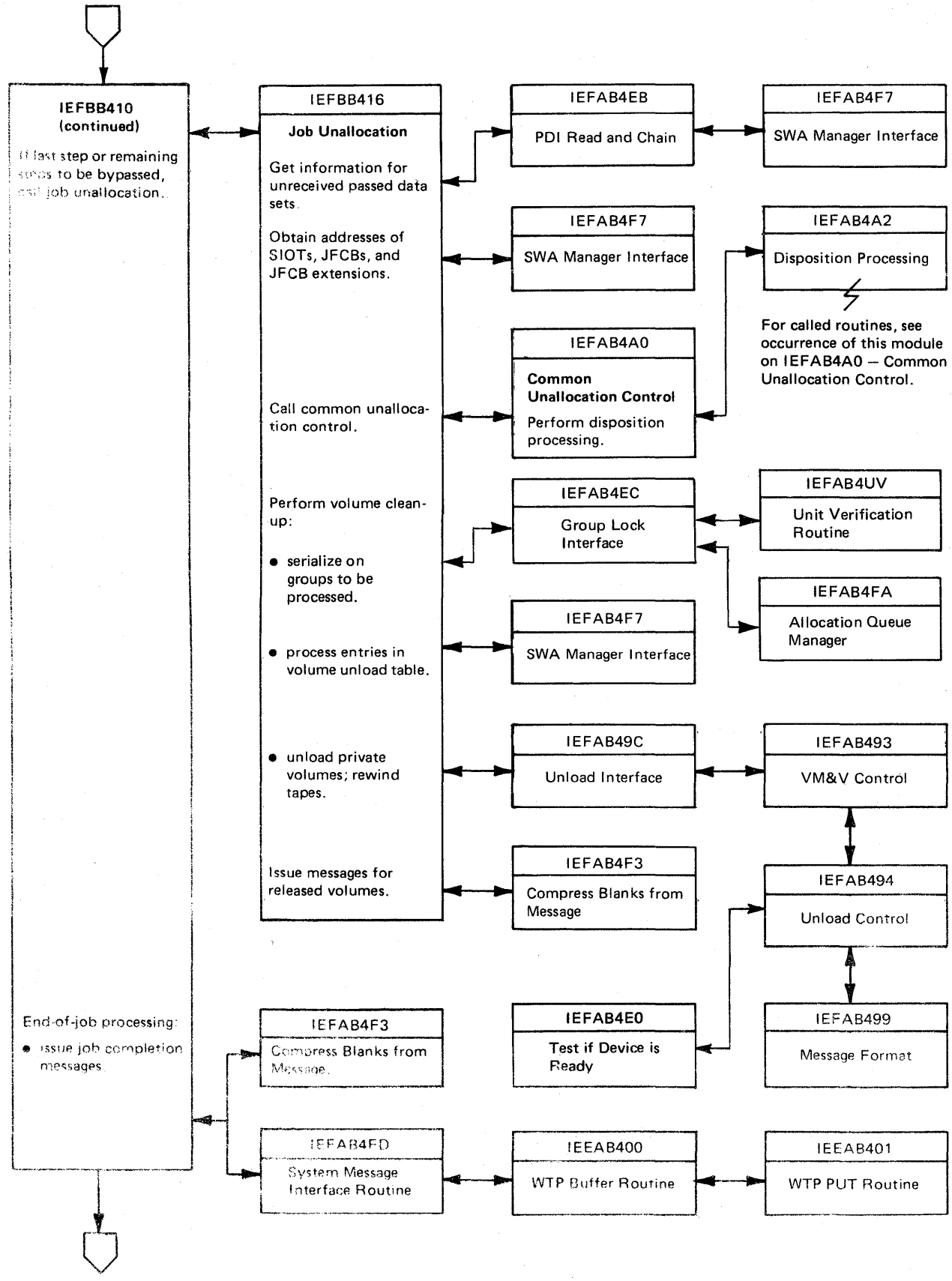


Figure 3-36. IEFBB410 – Initiator/Unallocation Interface (Part 2 of 3)

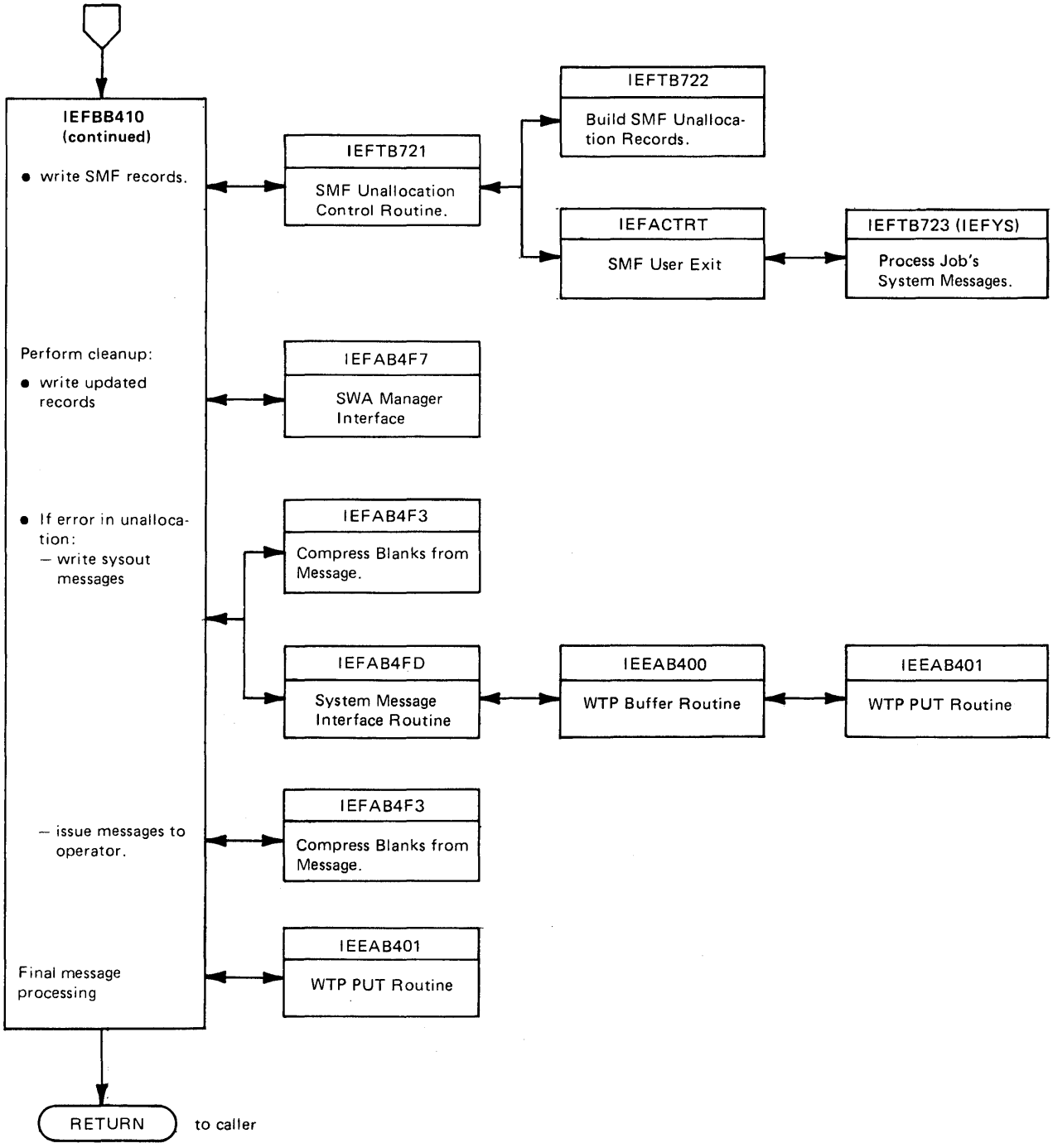
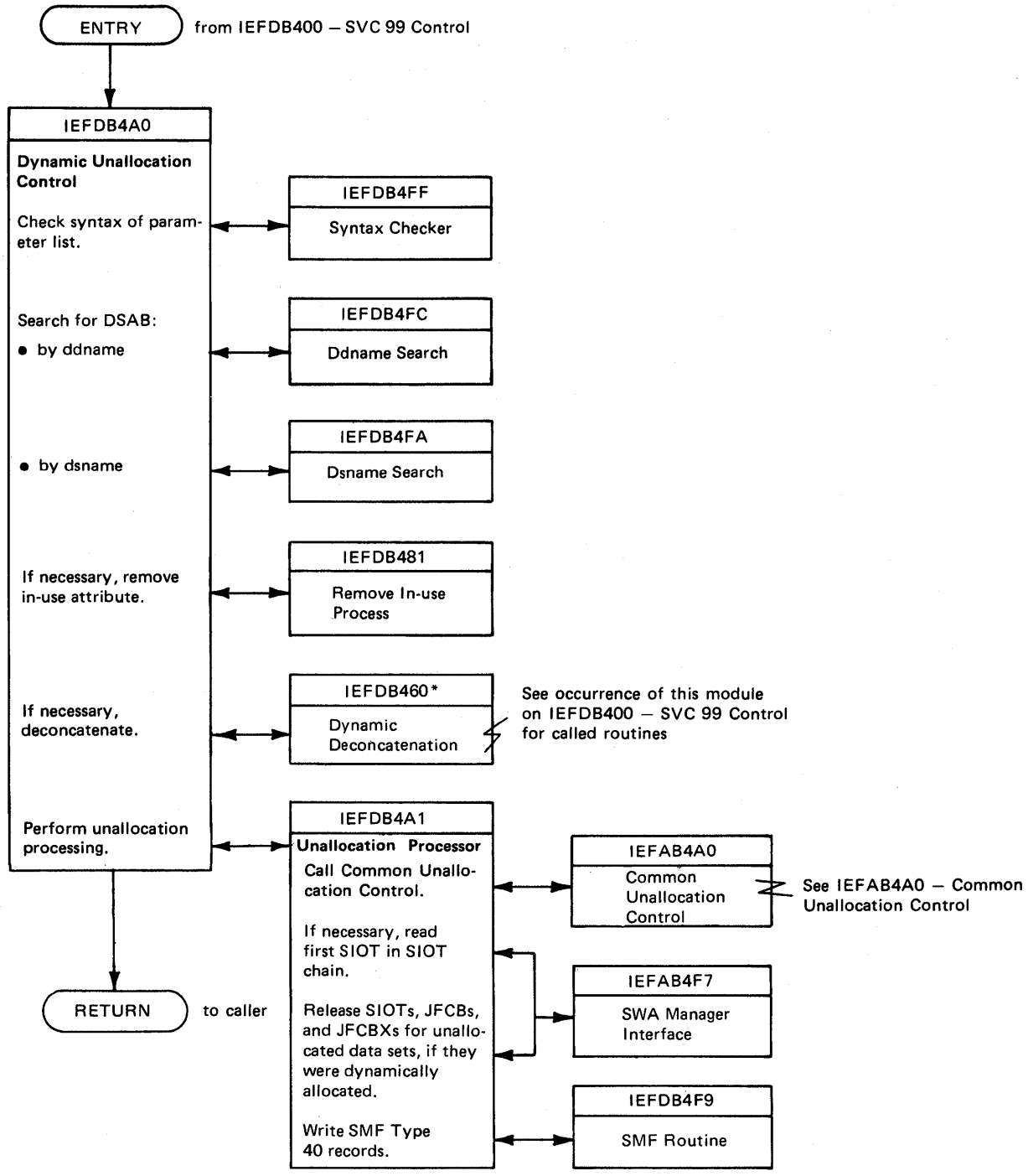


Figure 3-36. IEFBB410 – Initiator/Unallocation Interface (Part 3 of 3)

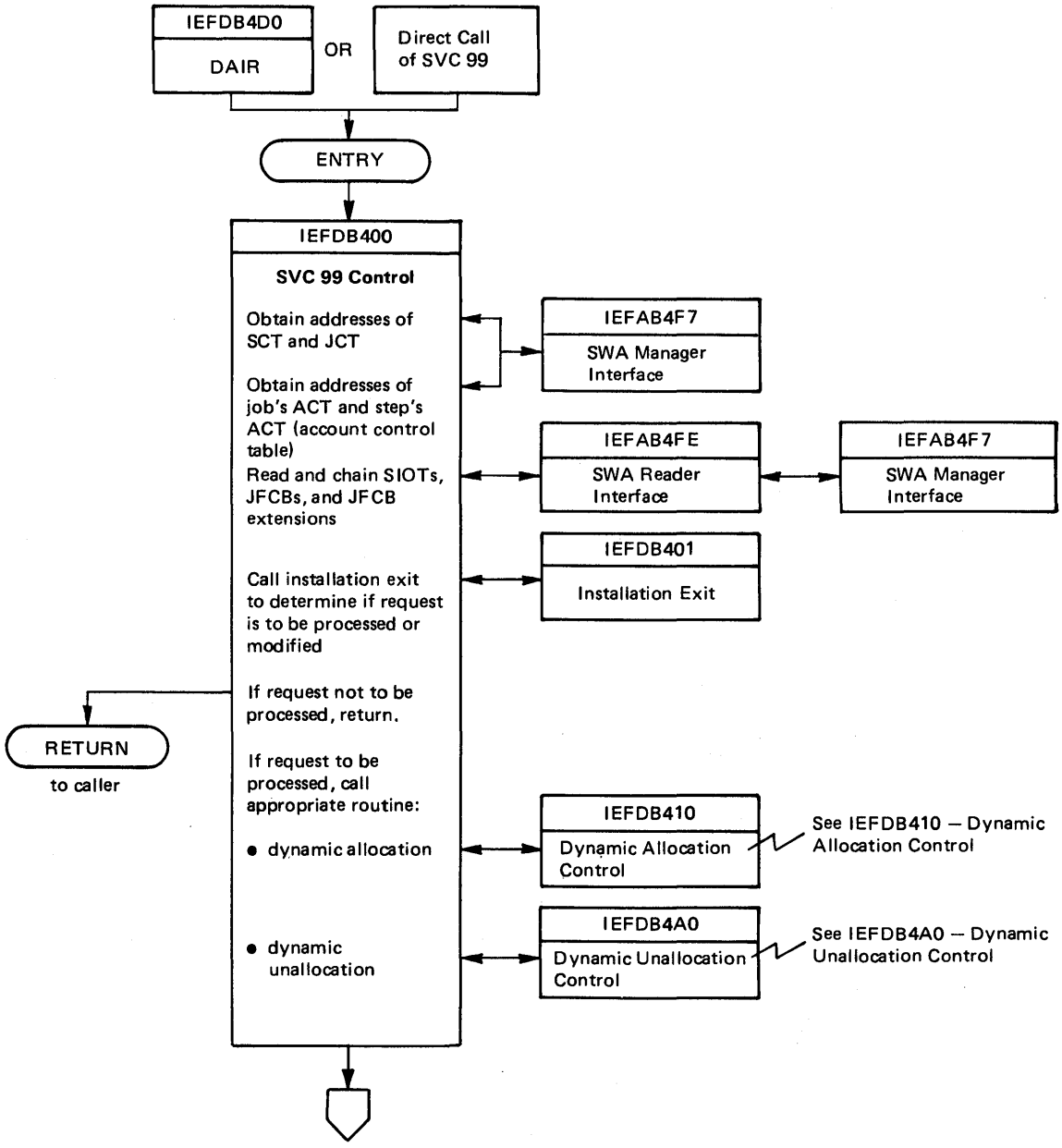


Notes:

1. An error in any routine causes return to the calling routine, except where noted.
2. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

*If the deconcatenation is unsuccessful, IEFDB4A0 will still unallocate other occurrences of this data set, if it is a multiply-allocated data set.

Figure 3-37. IEFDB4A0 – Dynamic Unallocation Control



Notes:

1. An error in any routine causes return to the calling routine.
2. All modules receive control by means of a CALL instruction in PLS, which generates a BALR instruction in assembler language.

Figure 3-38. IEFDB400 – SVC 99 Control (Part 1 of 3)

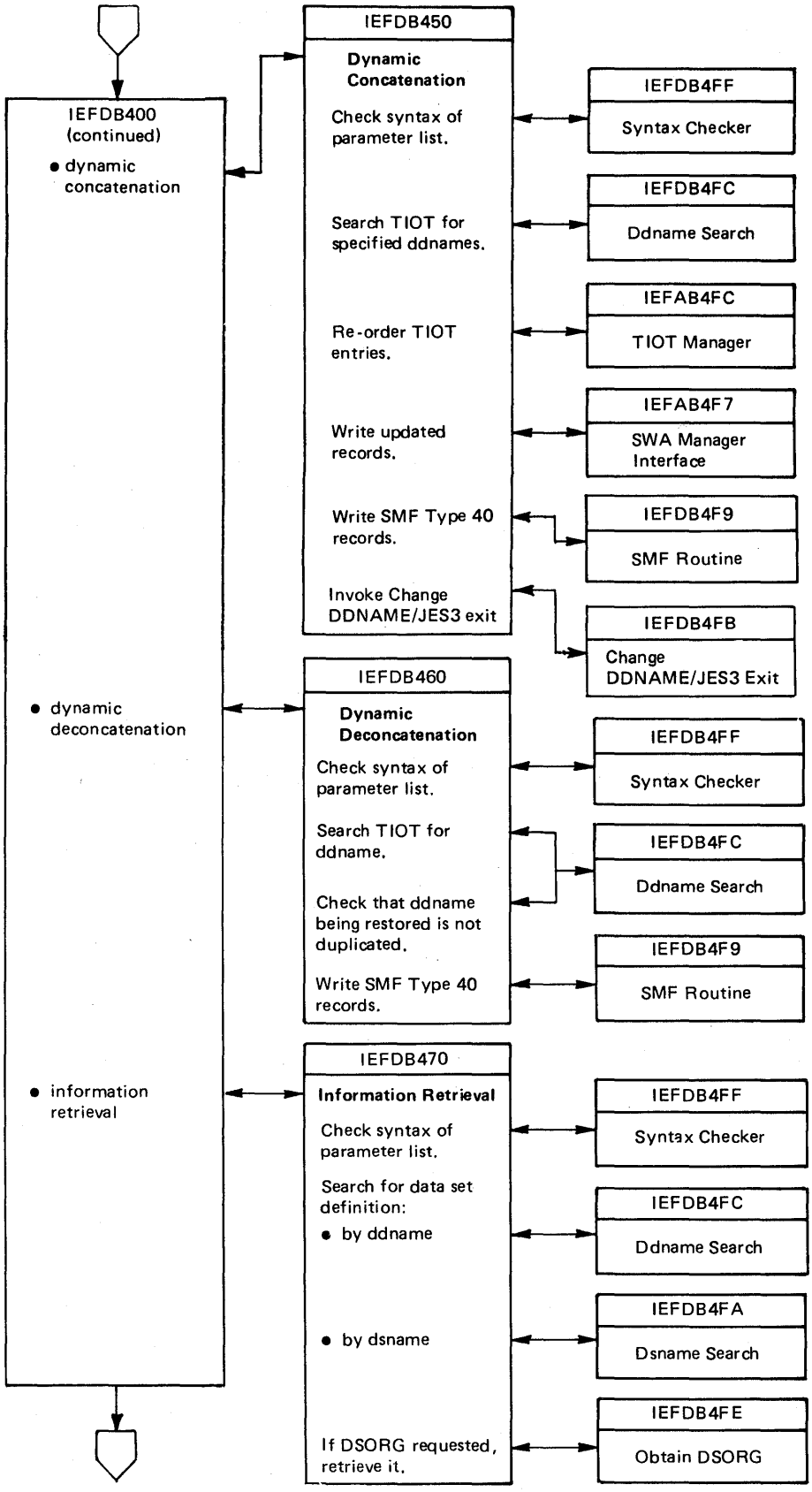


Figure 3-38. IEFDB400 – SVC 99 Control (Part 2 of 3)

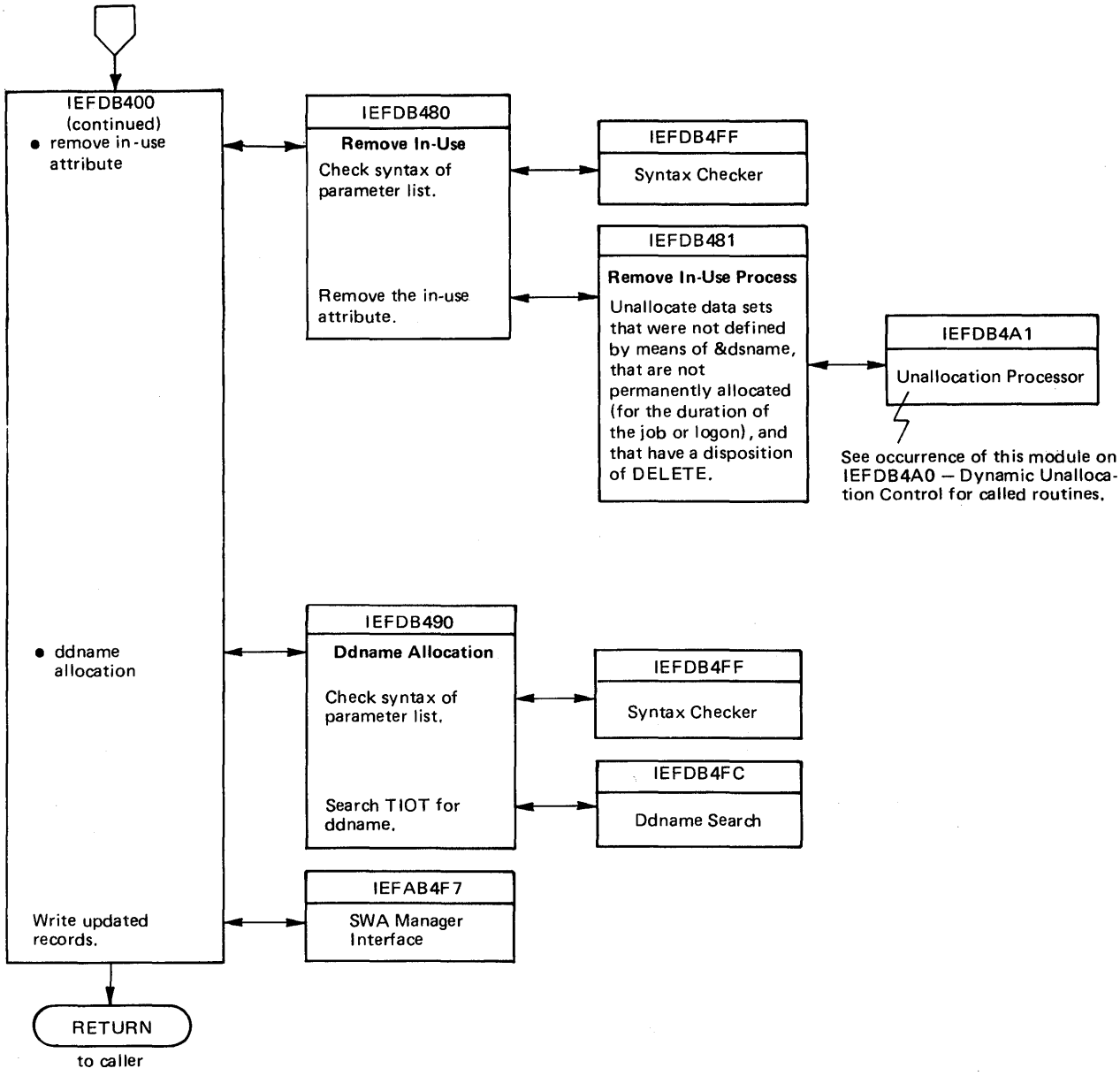


Figure 3-38. IEFDB400 – SVC 99 Control (Part 3 of 3)

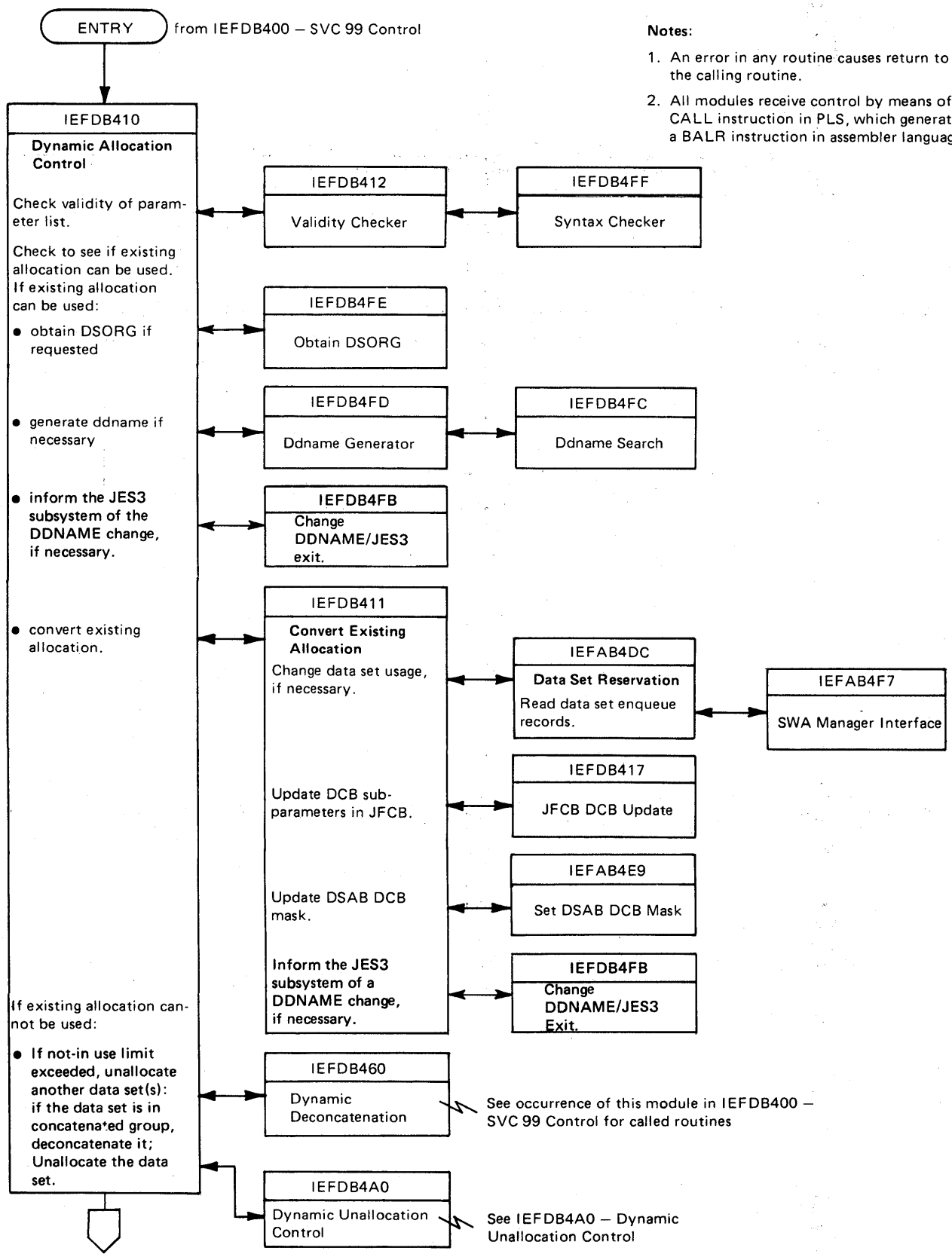
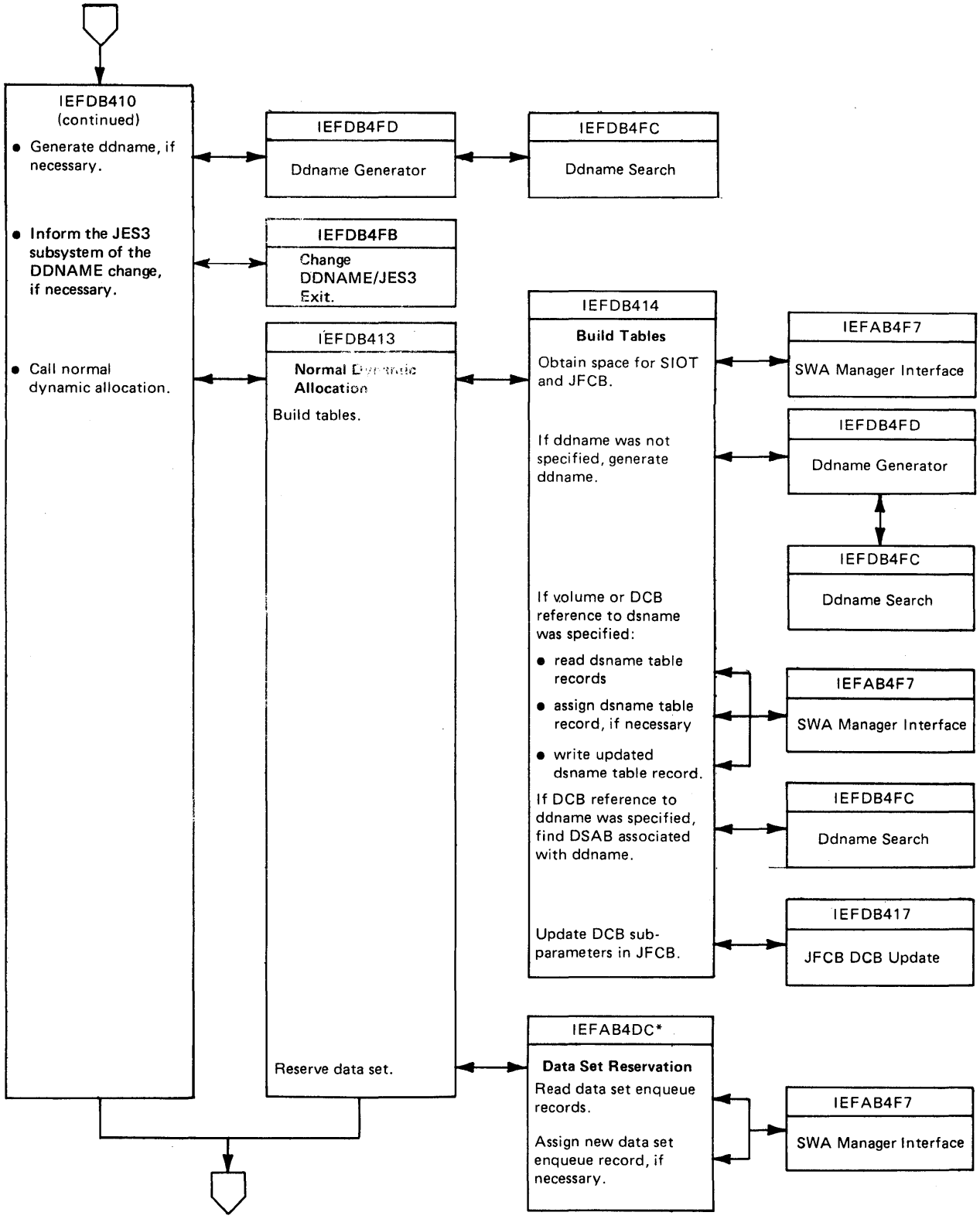


Figure 3-39. IEFDB410 – Dynamic Allocation Control (Part 1 of 4)



***Note:** If a data set enqueue record exists for the data set as SHR, but this request specifies exclusive, the record and type of enqueue is not changed until the allocation is successfully performed.

Figure 3-39. IEFDB410 – Dynamic Allocation Control (Part 2 of 4)

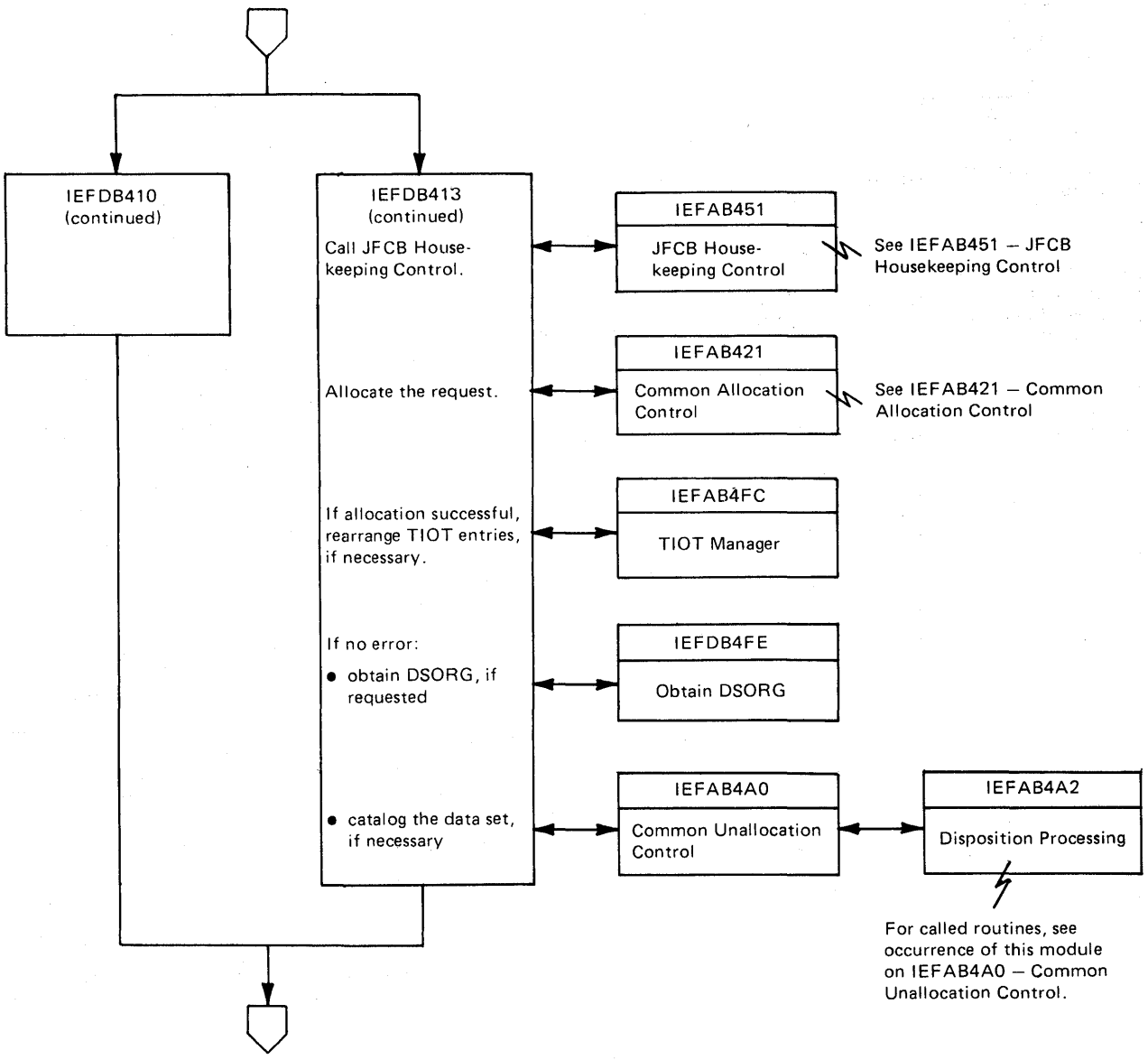


Figure 3-39. IEFDB410 – Dynamic Allocation Control (Part 3 of 4)

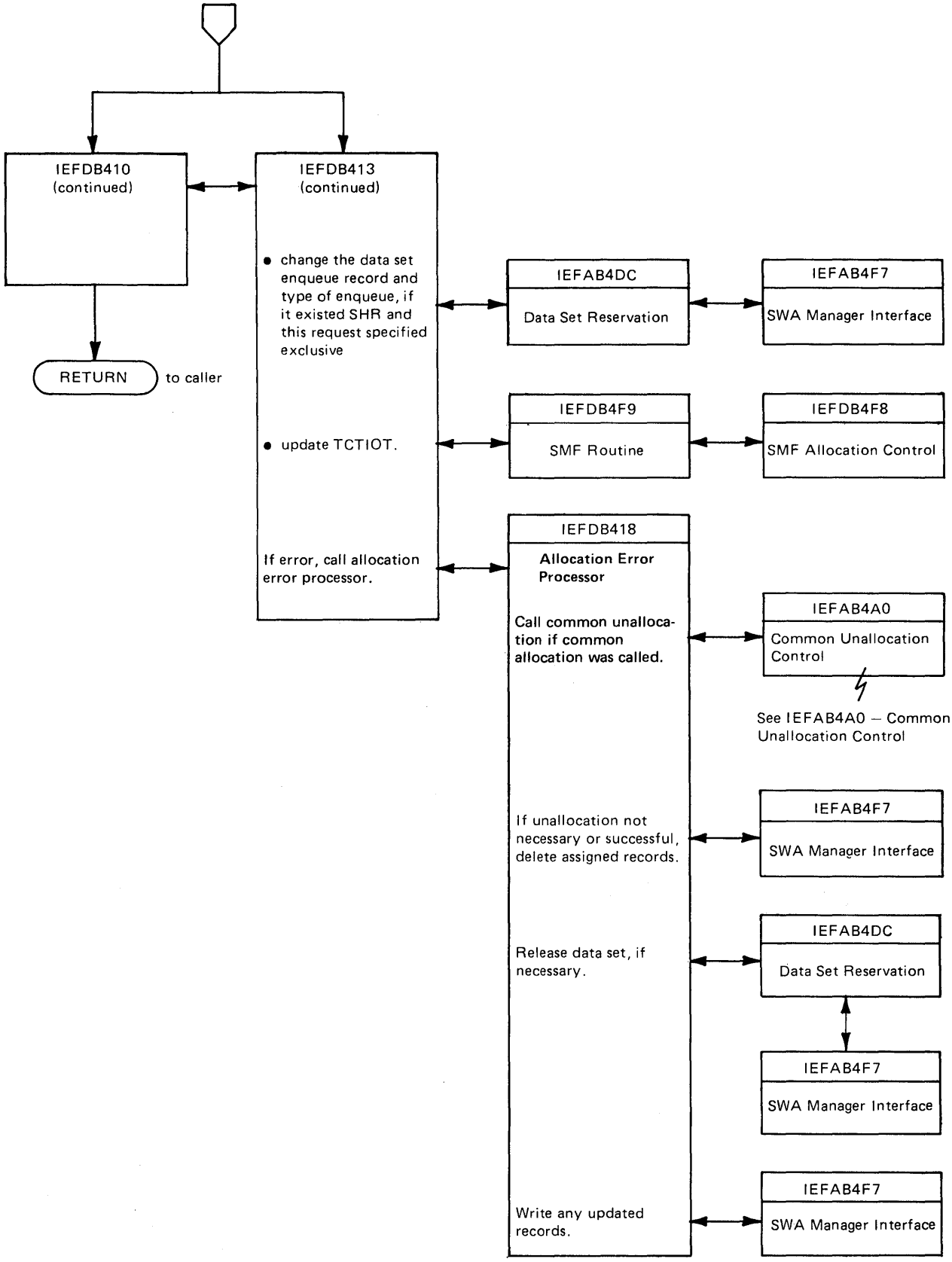
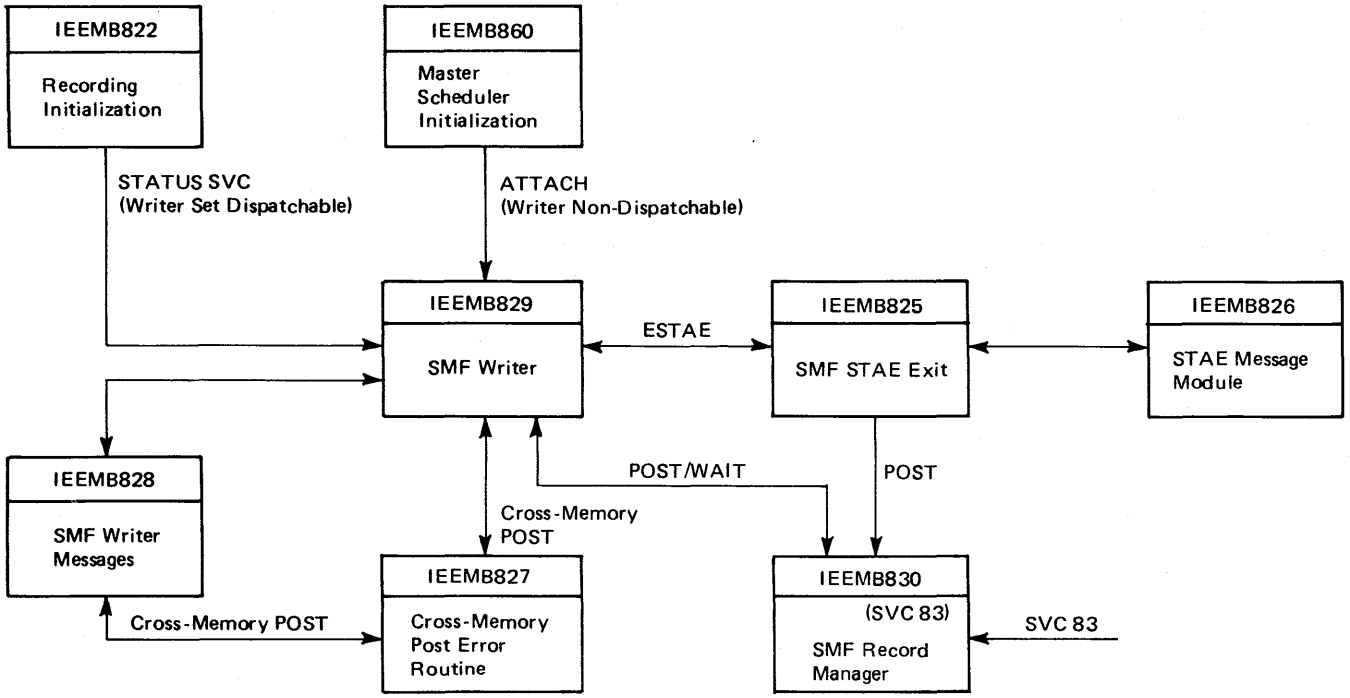


Figure 3-39. IEFDB410 – Dynamic Allocation Control (Part 4 of 4)



Note:
All modules except IEEMB827 and IEEMB830 perform their processing in the master (scheduler) storage.

Figure 3-40. System Management Facilities (SMF) Recording: Program Organization

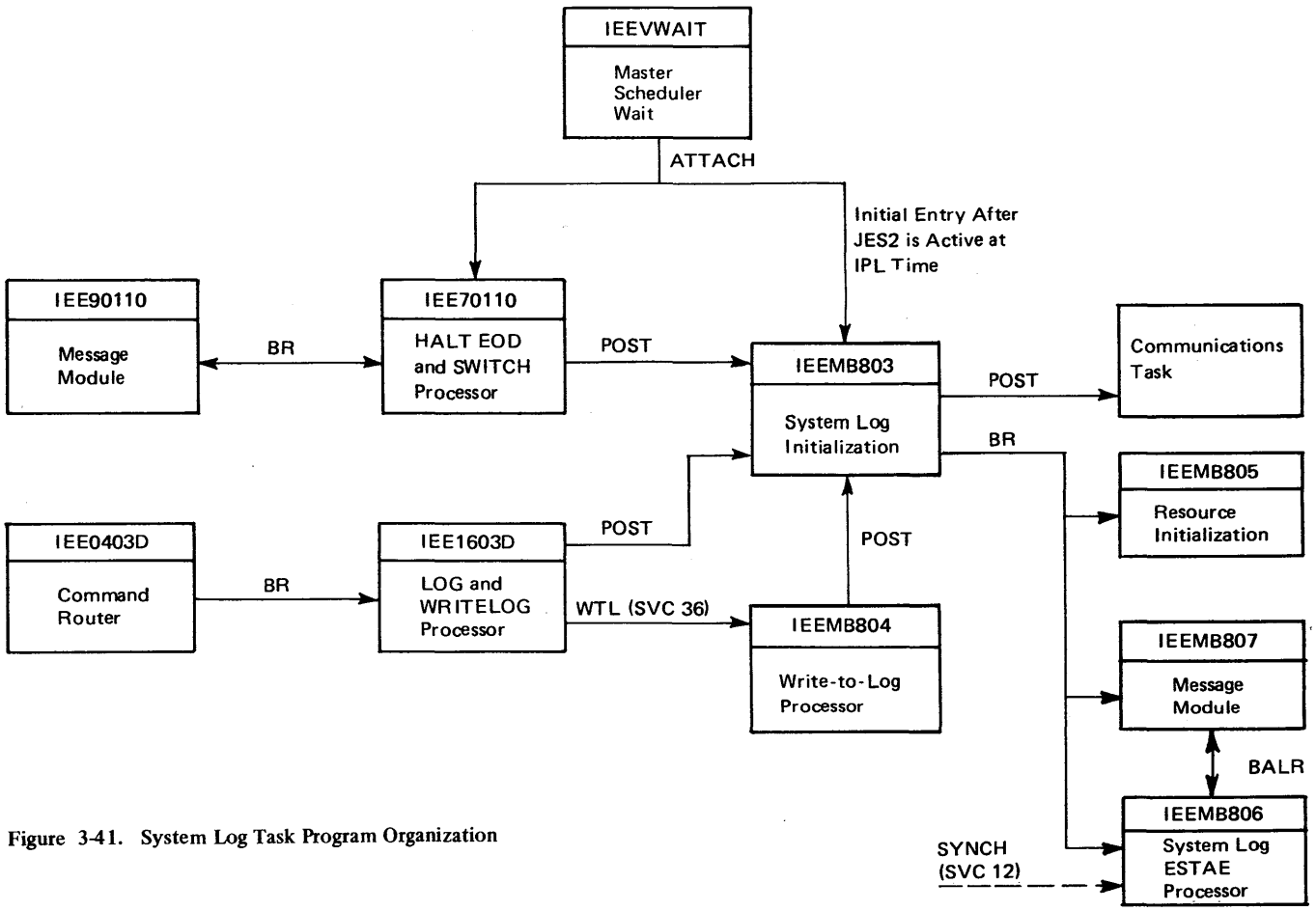


Figure 3-41. System Log Task Program Organization

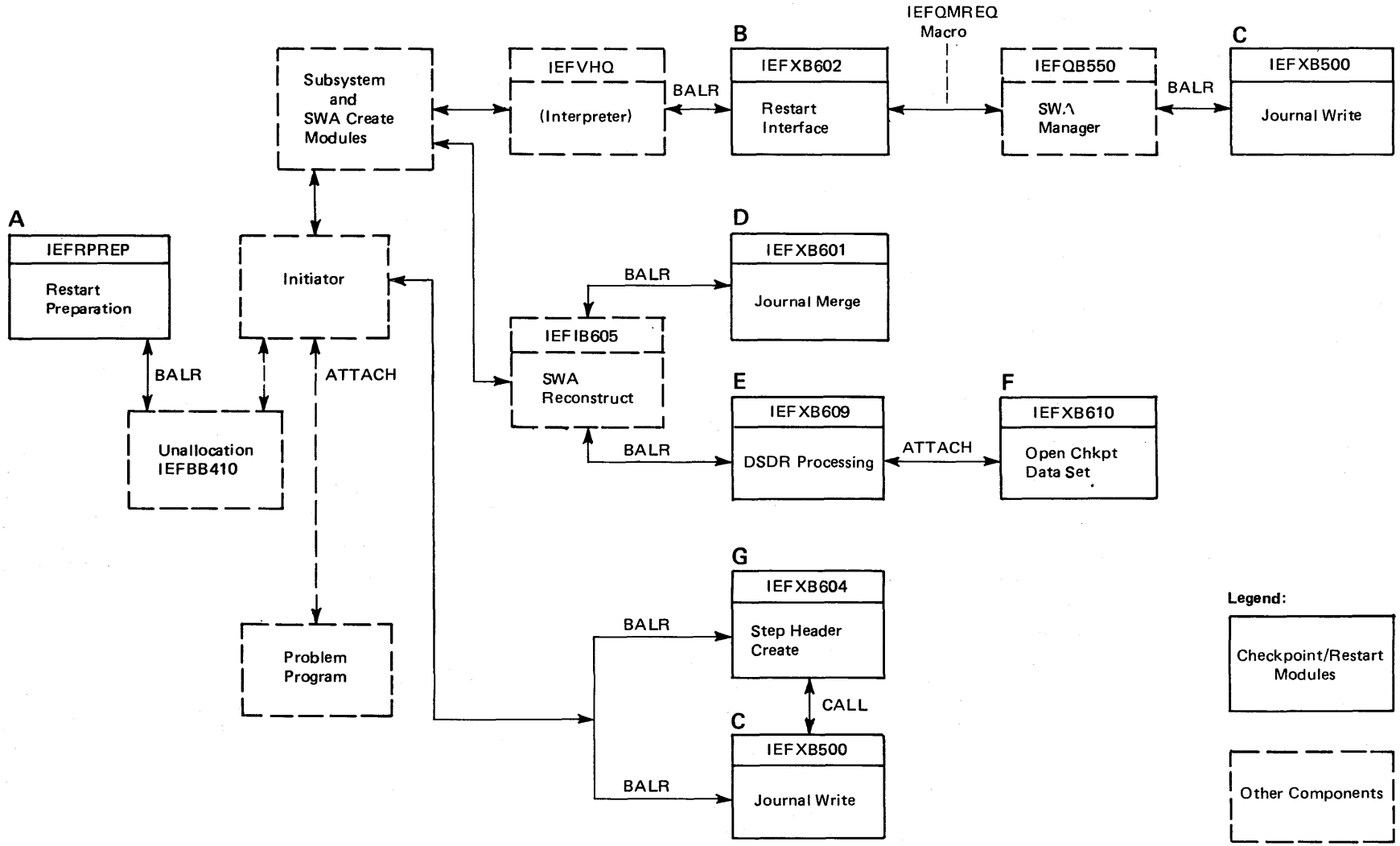


Figure 3-42. Checkpoint/Restart Program Organization

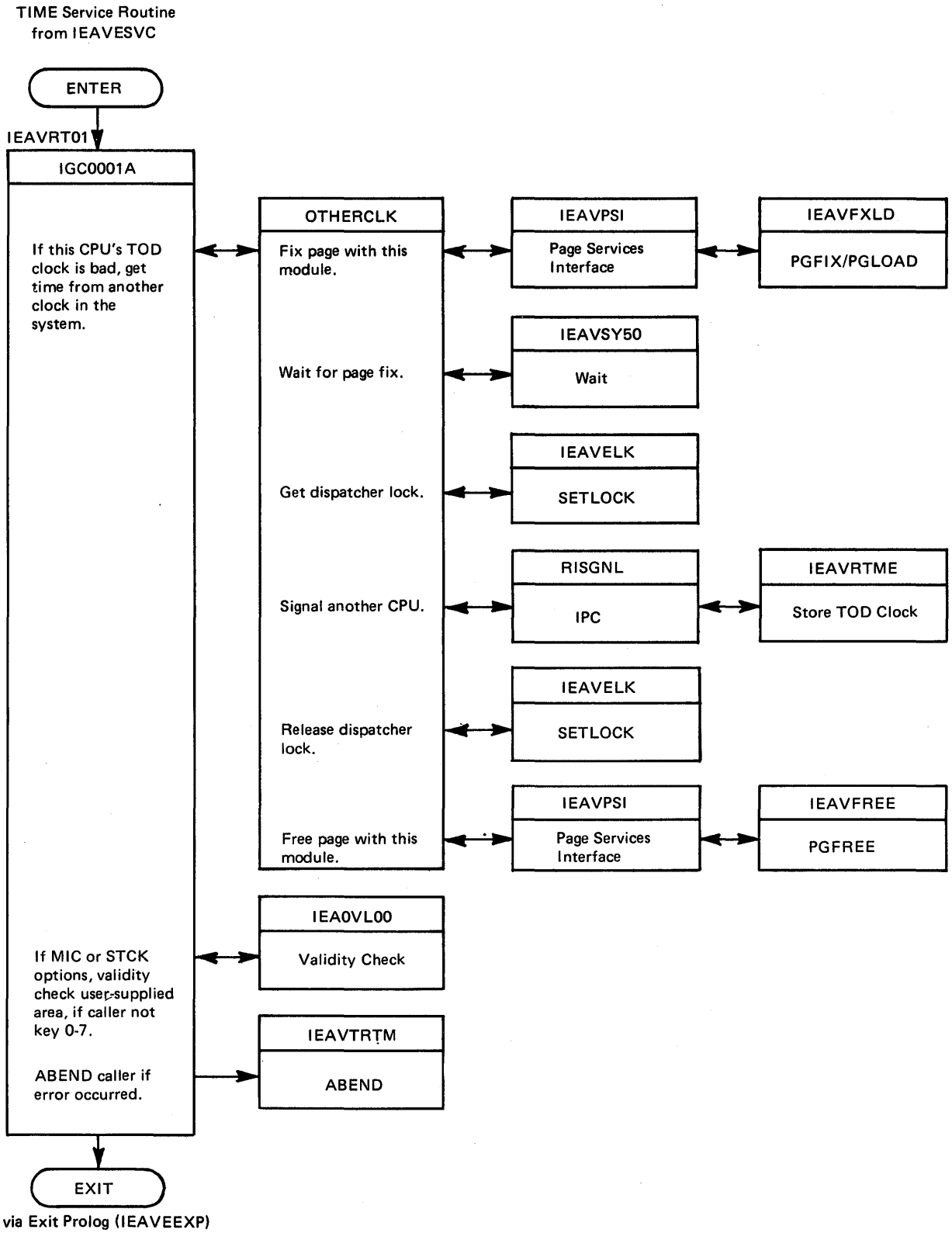


Figure 3-43. Timing Services Module Flow (Parts 1-2 of 23)

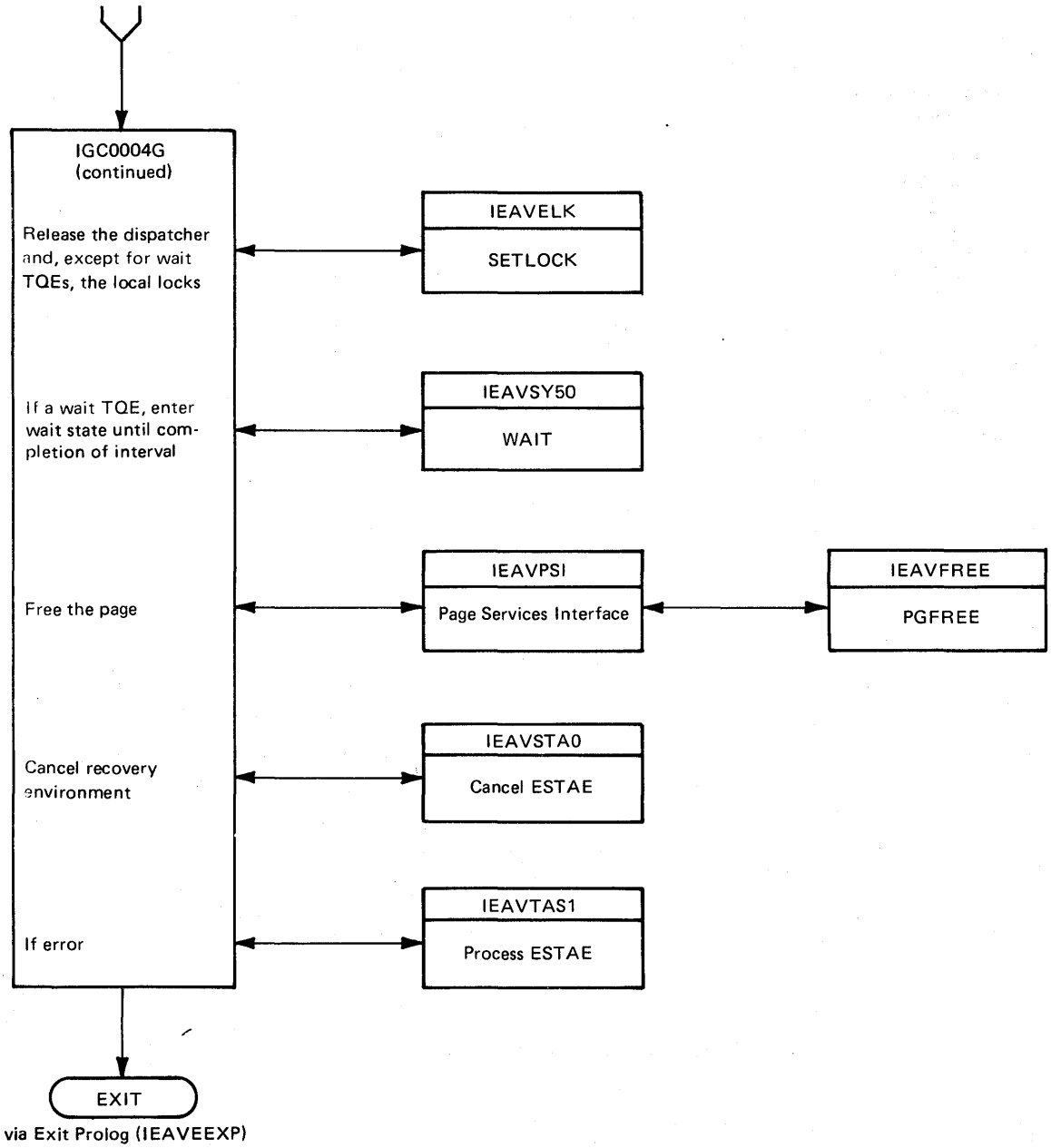


Figure 3-43. Timing Services Module Flow (Part 5 of 23)

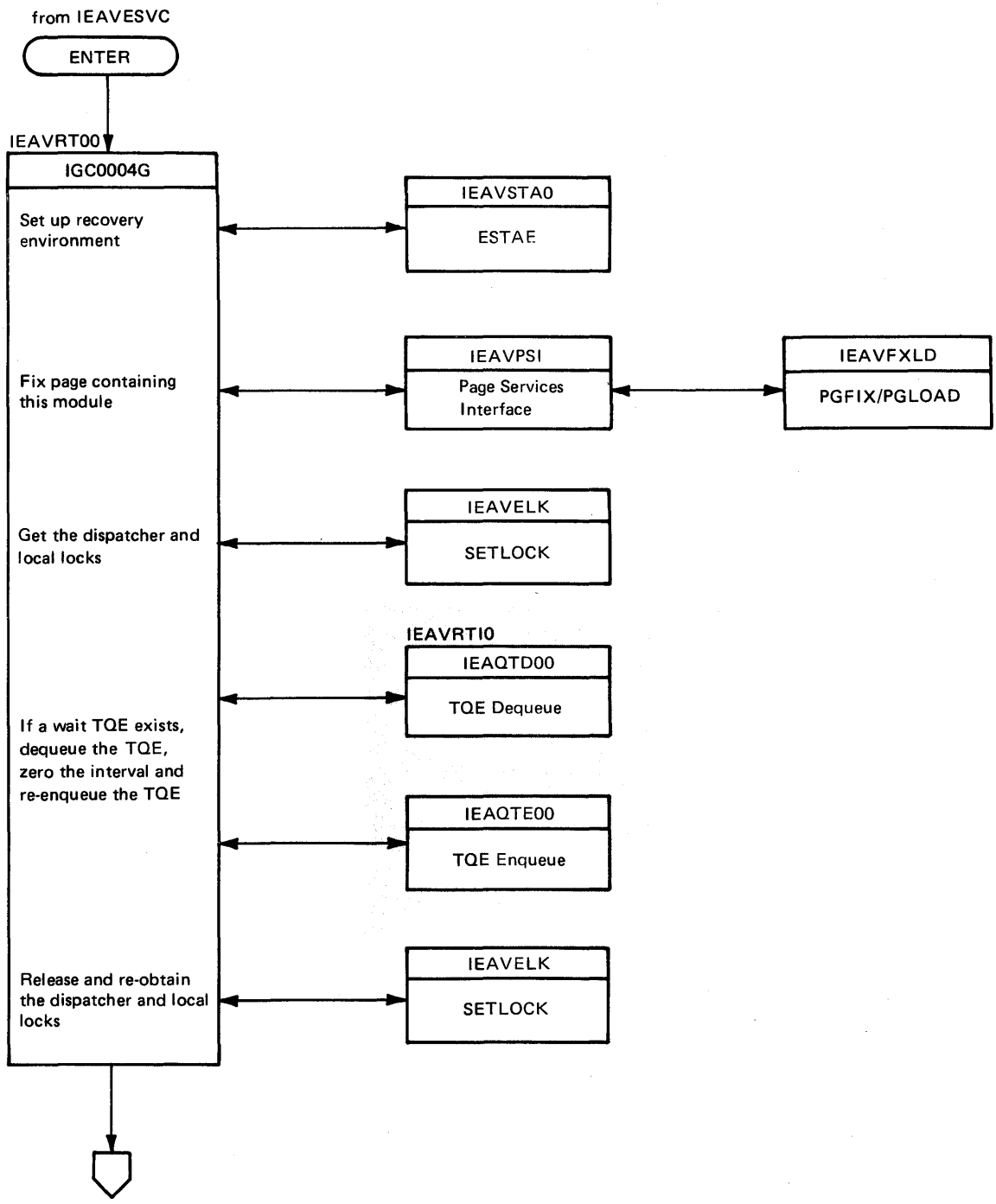


Figure 3-43. Timing Services Module Flow (Part 3 of 23)

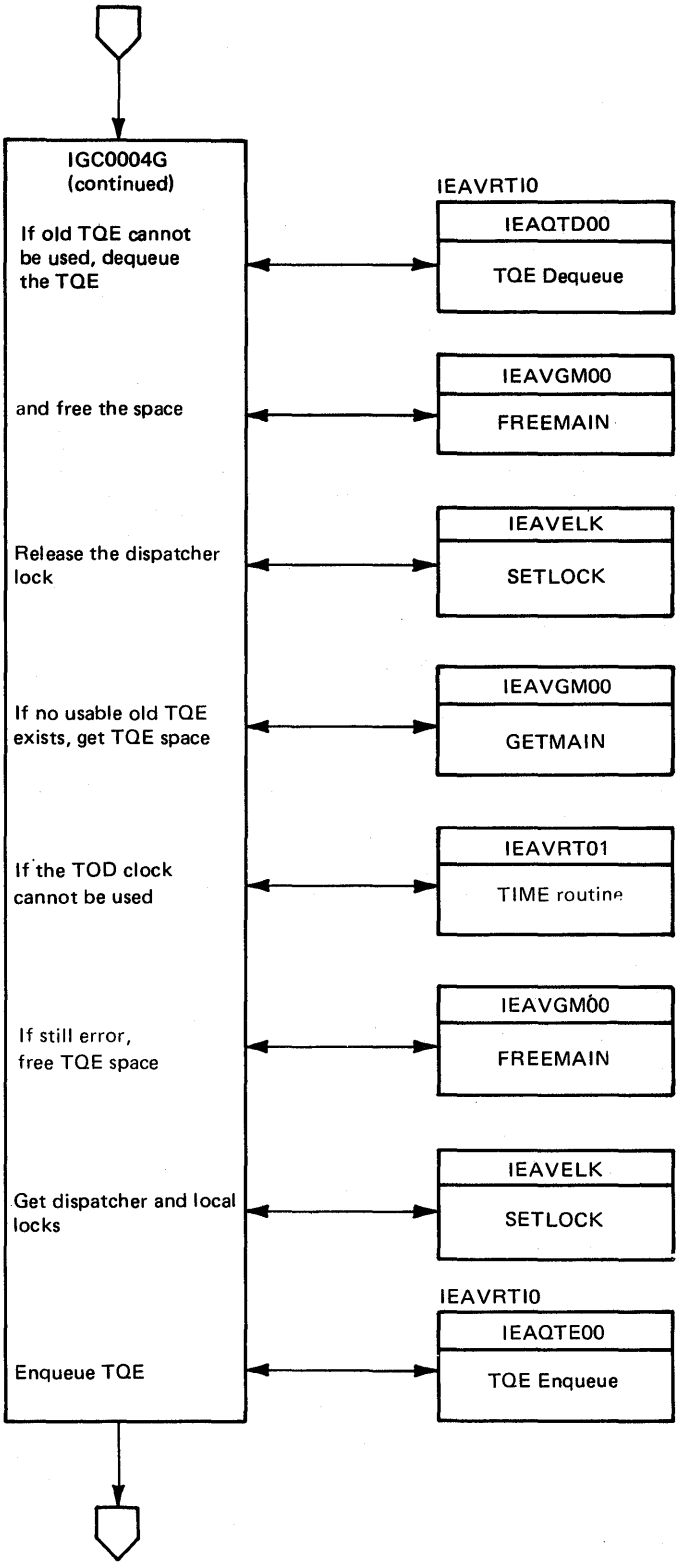


Figure 3-43. Timing Services Module Flow (Part 4 of 23)

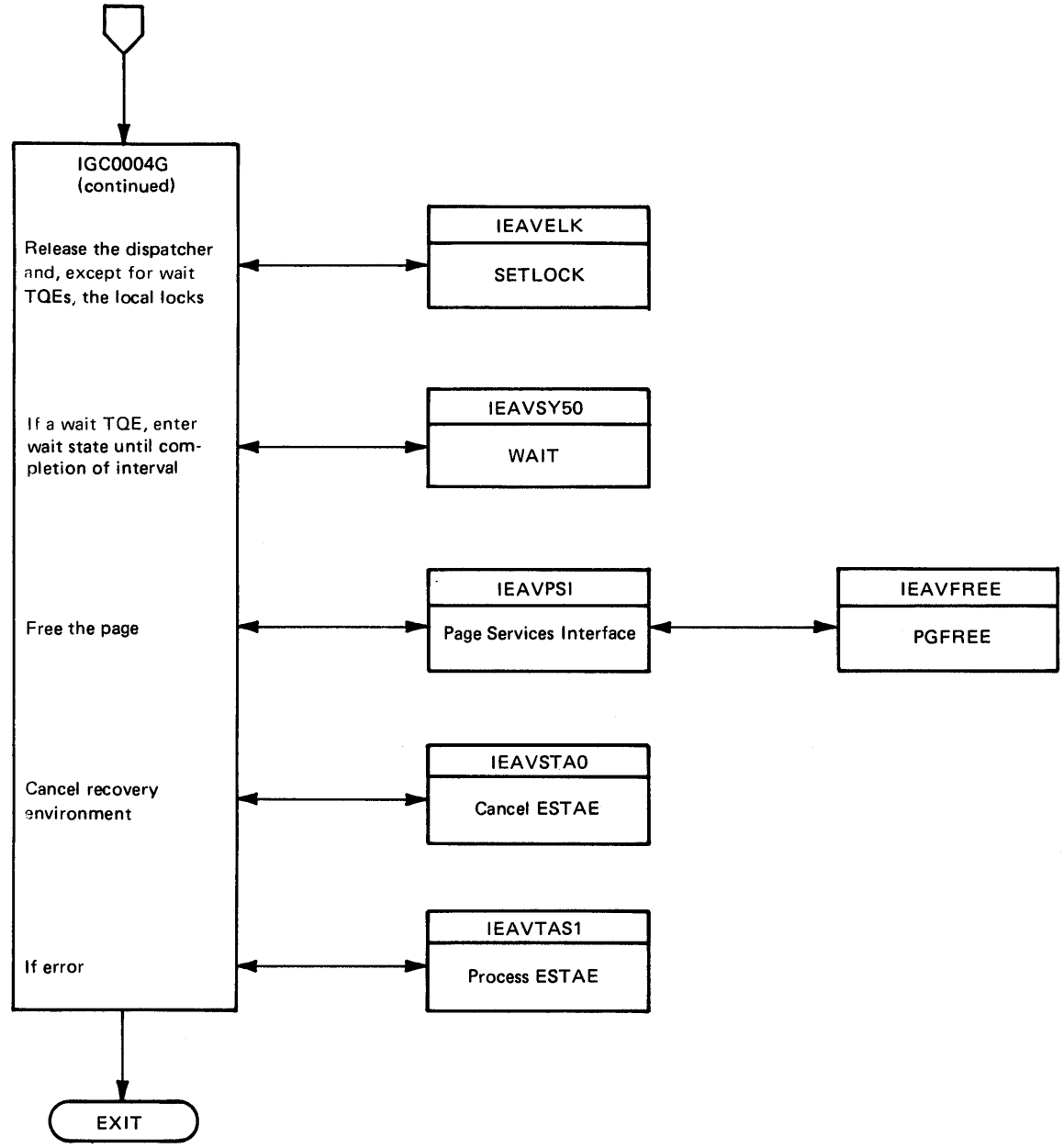


Figure 3-43. Timing Services Module Flow (Part 5 of 23)

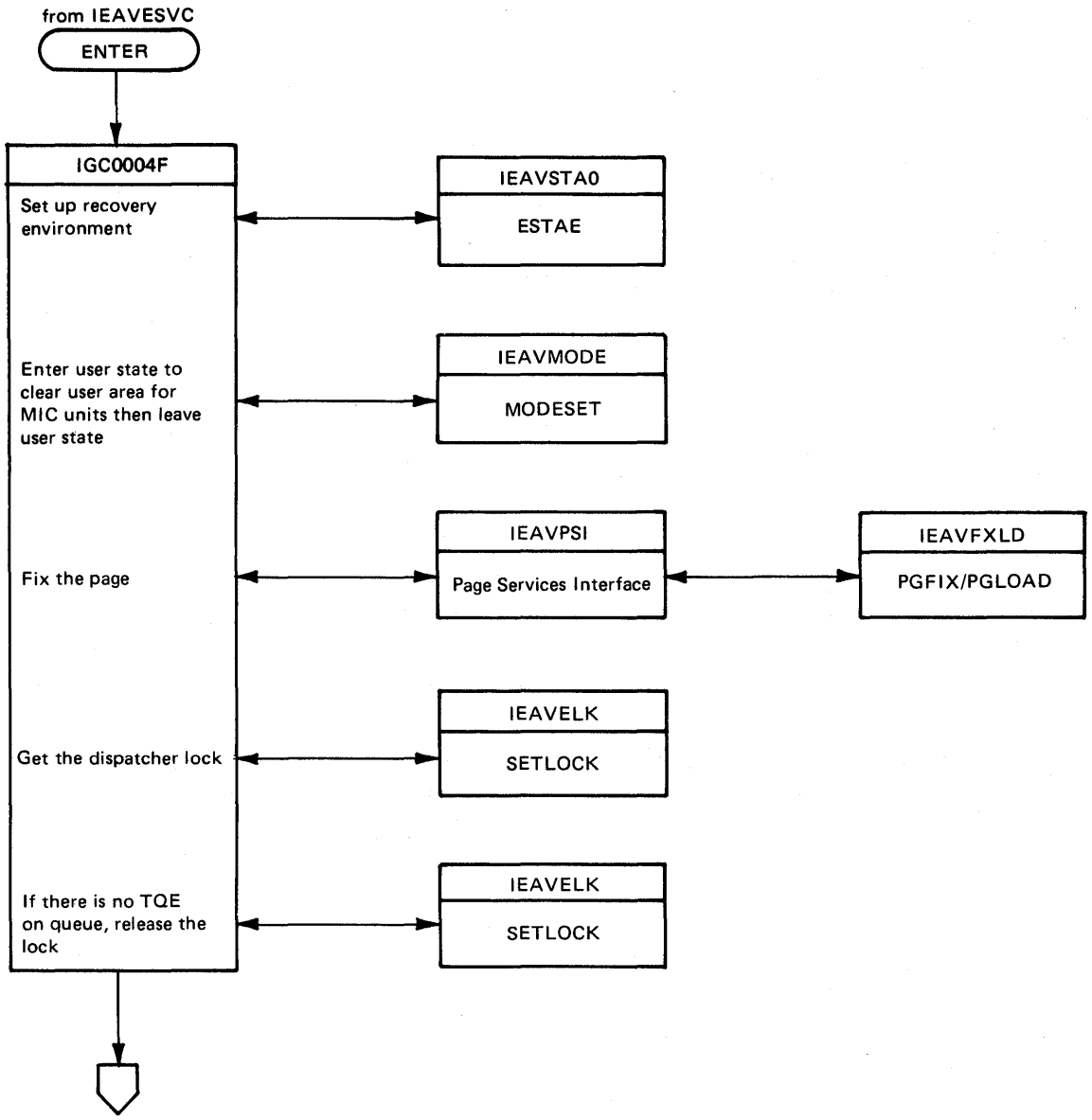


Figure 3-43. Timing Services Module Flow (Part 6 of 23)

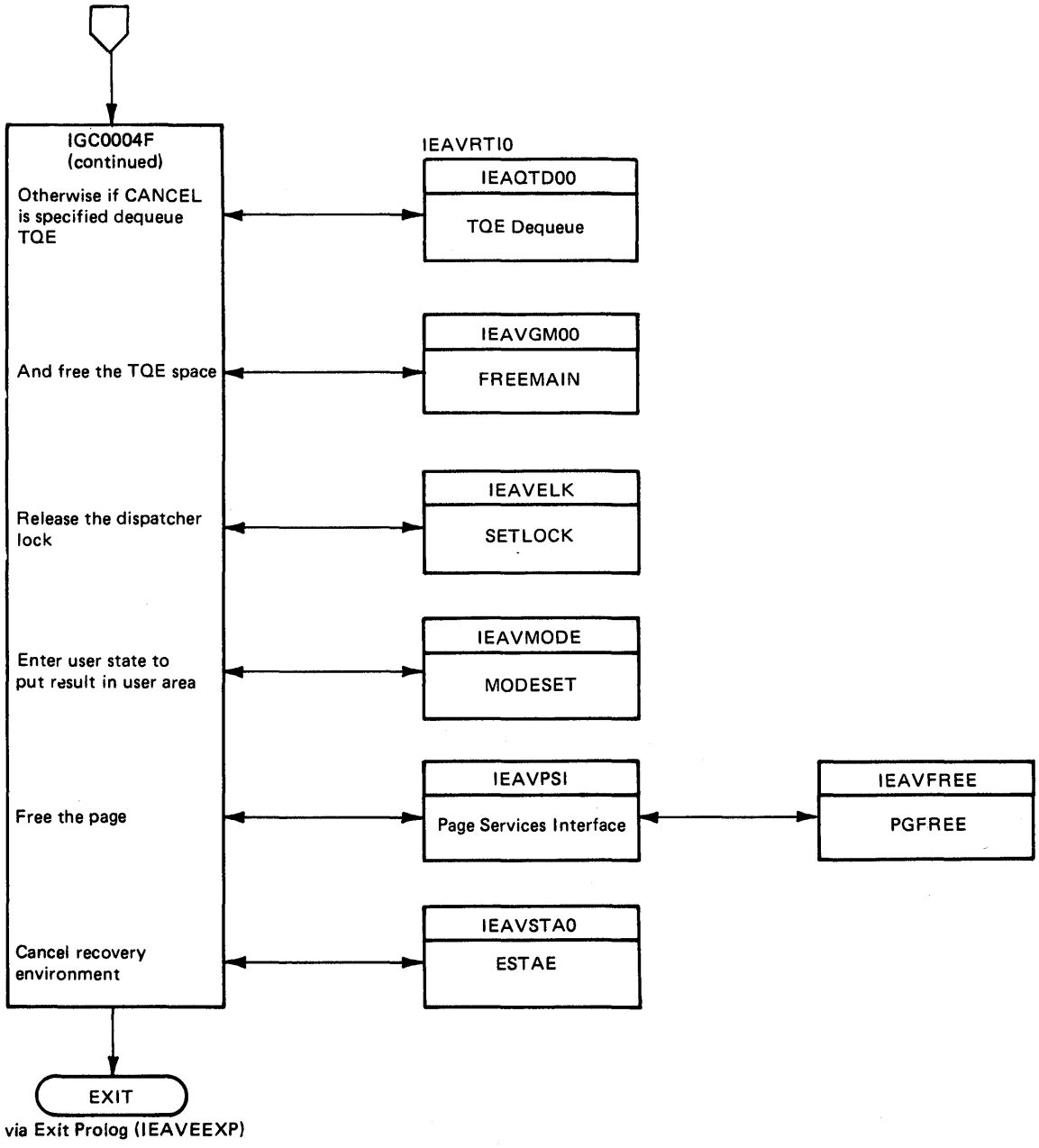


Figure 3-43. Timing Services Module Flow (Part 7 of 23)

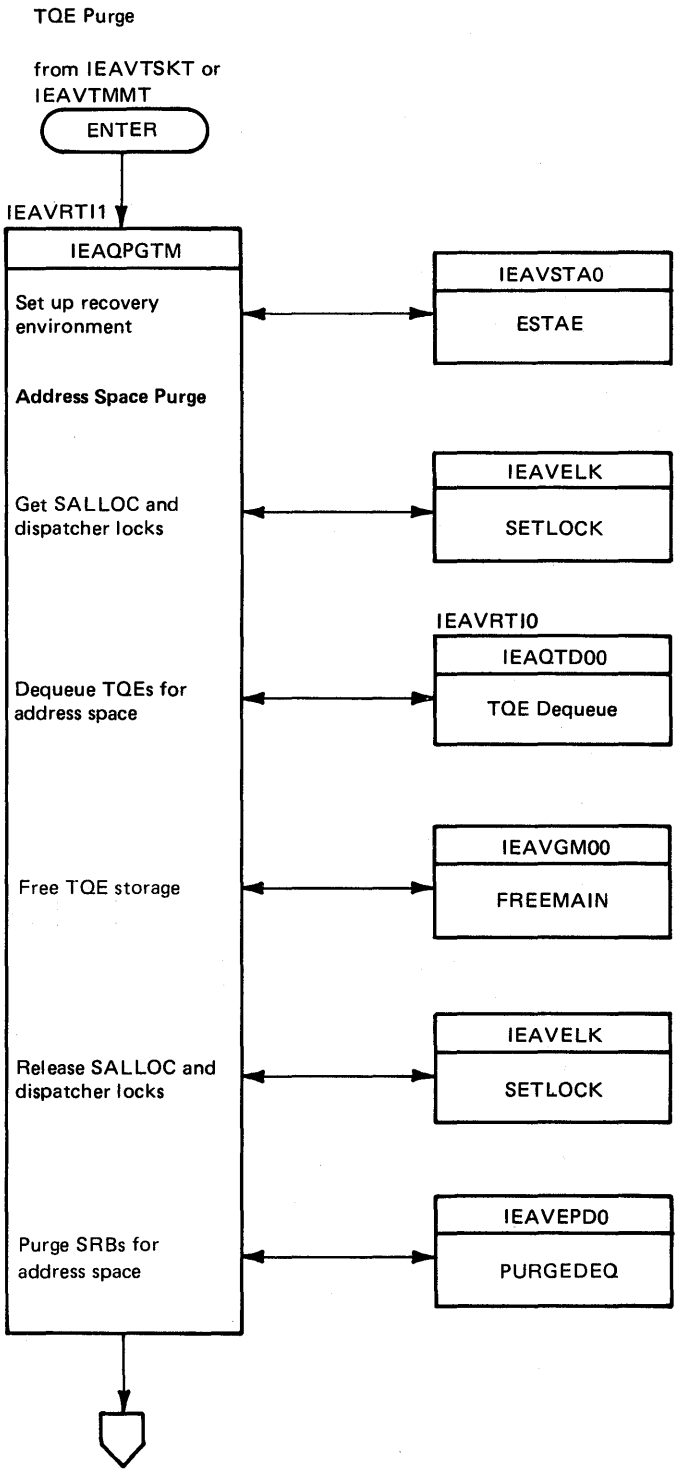


Figure 3-43. Timing Services Module Flow (Part 8 of 23)

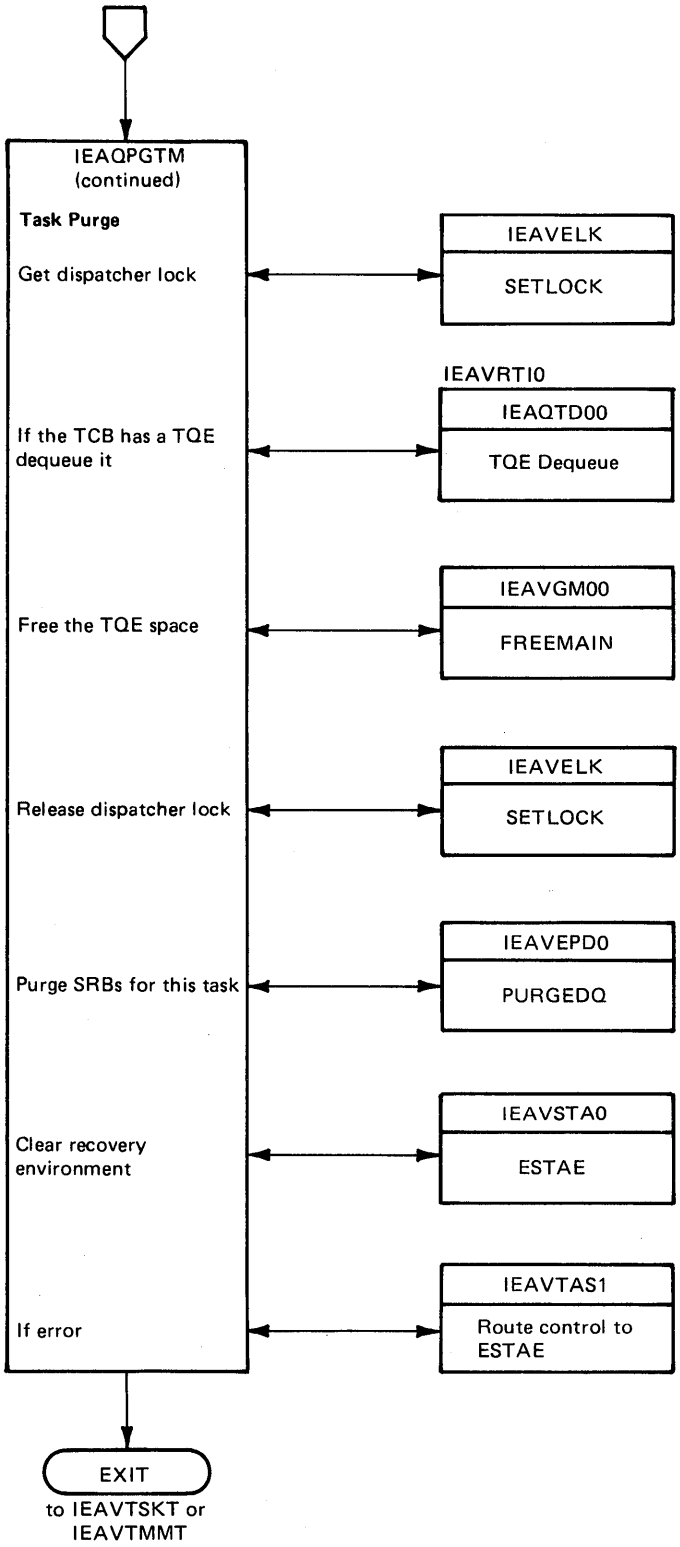


Figure 3-43. Timing Services Module Flow (Part 9 of 23)

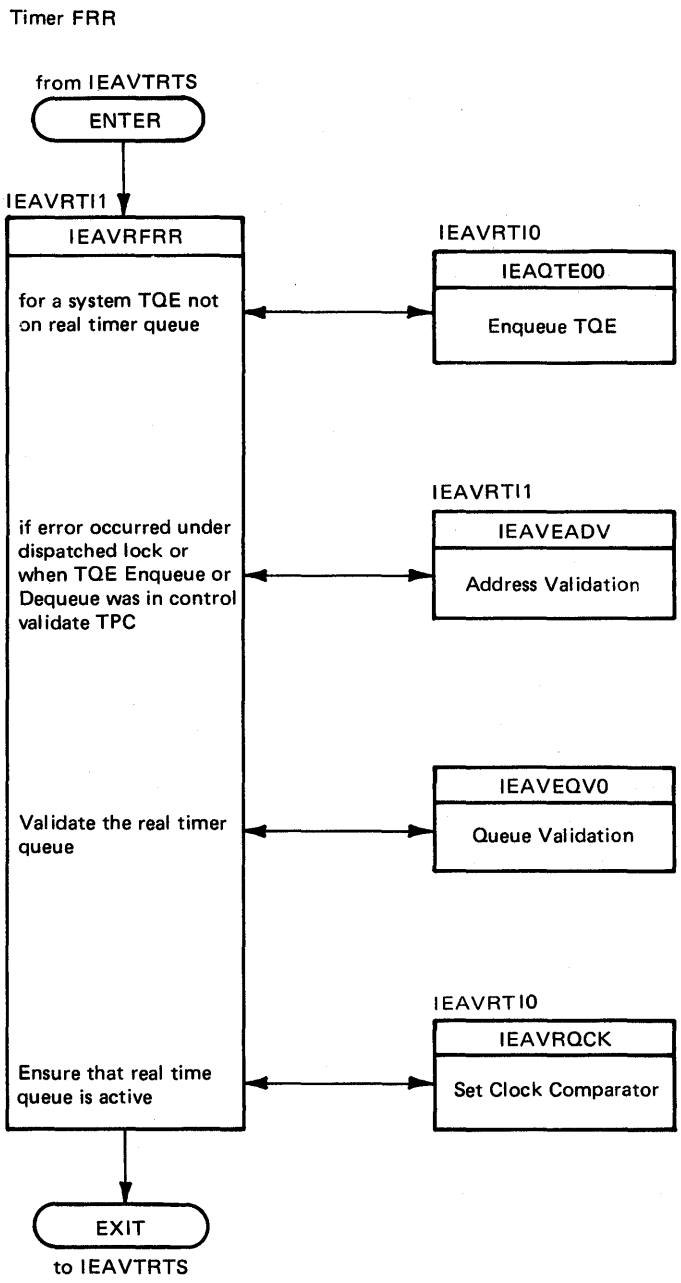


Figure 3-43. Timing Services Module Flow (Part 10 of 23)

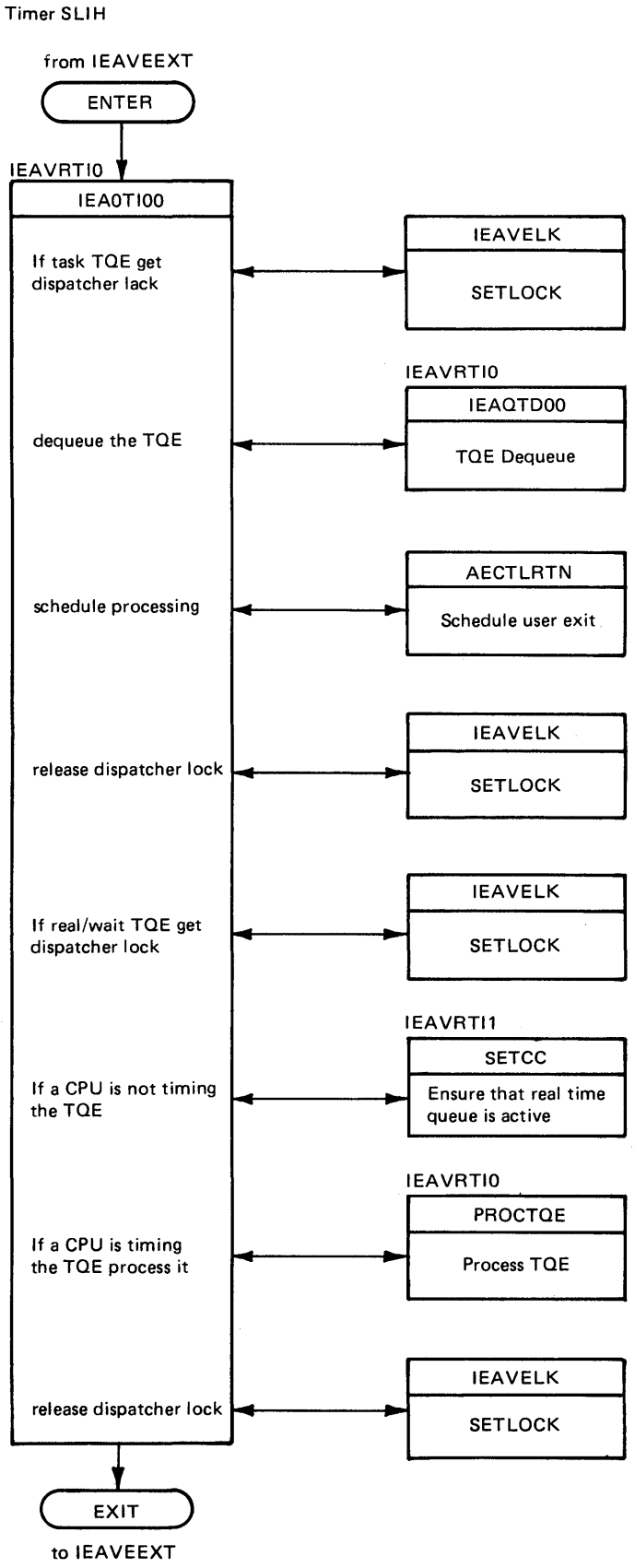


Figure 3-43. Timing Services Module Flow (Part 11 of 23)

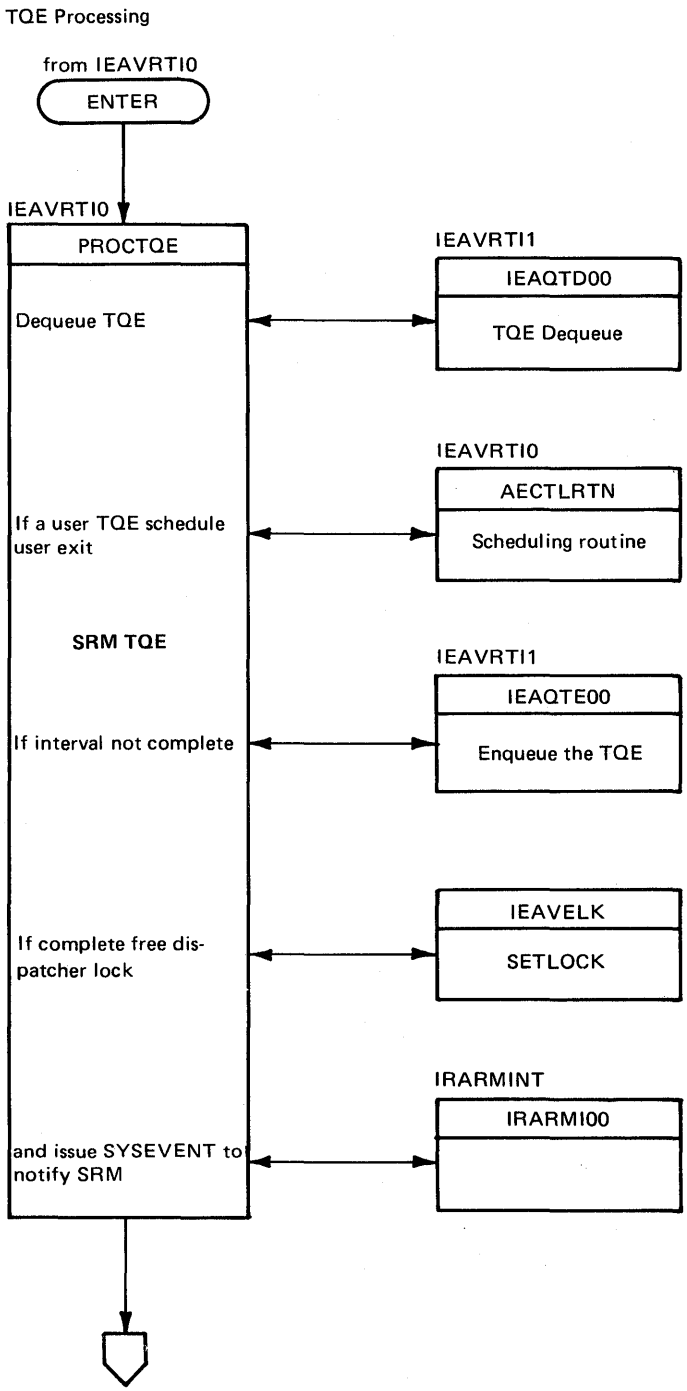


Figure 3-43. Timing Services Module Flow (Part 12 of 23)

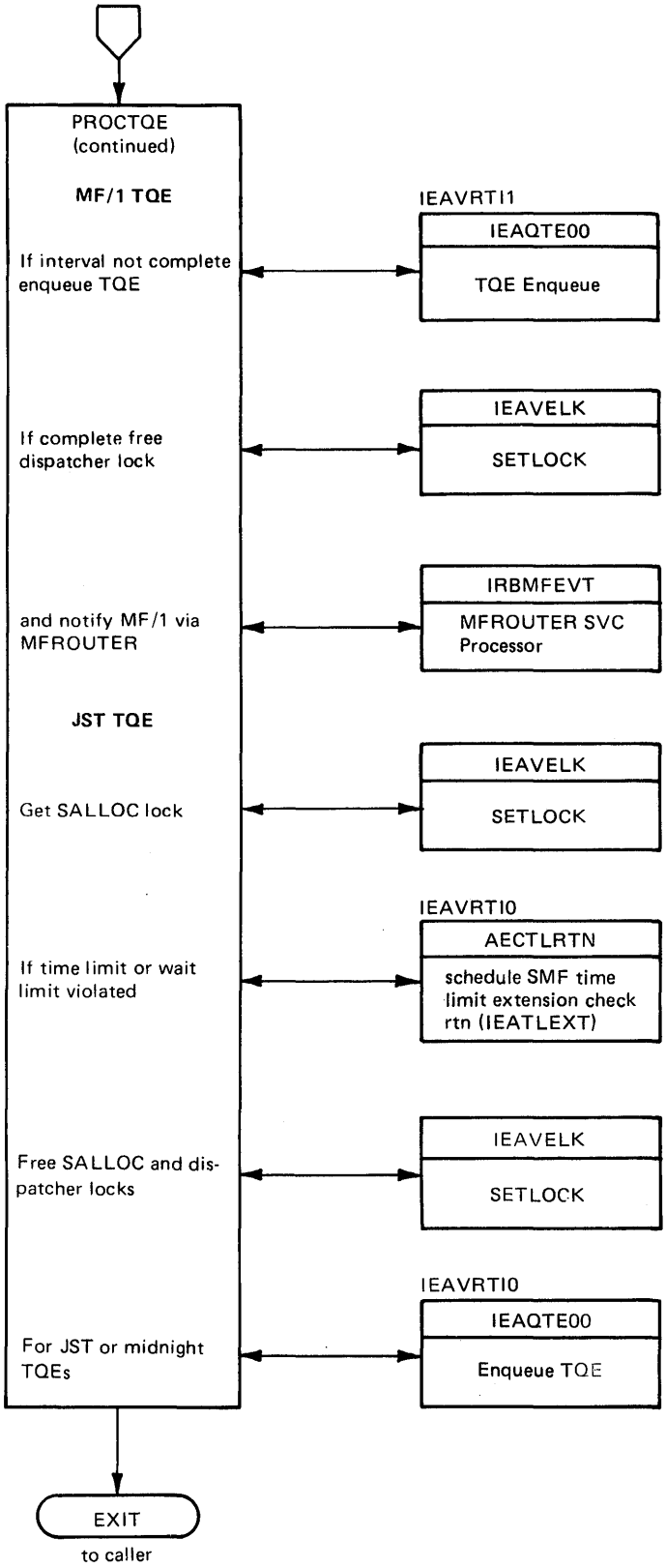


Figure 3-43. Timing Services Module Flow (Part 13 of 23)

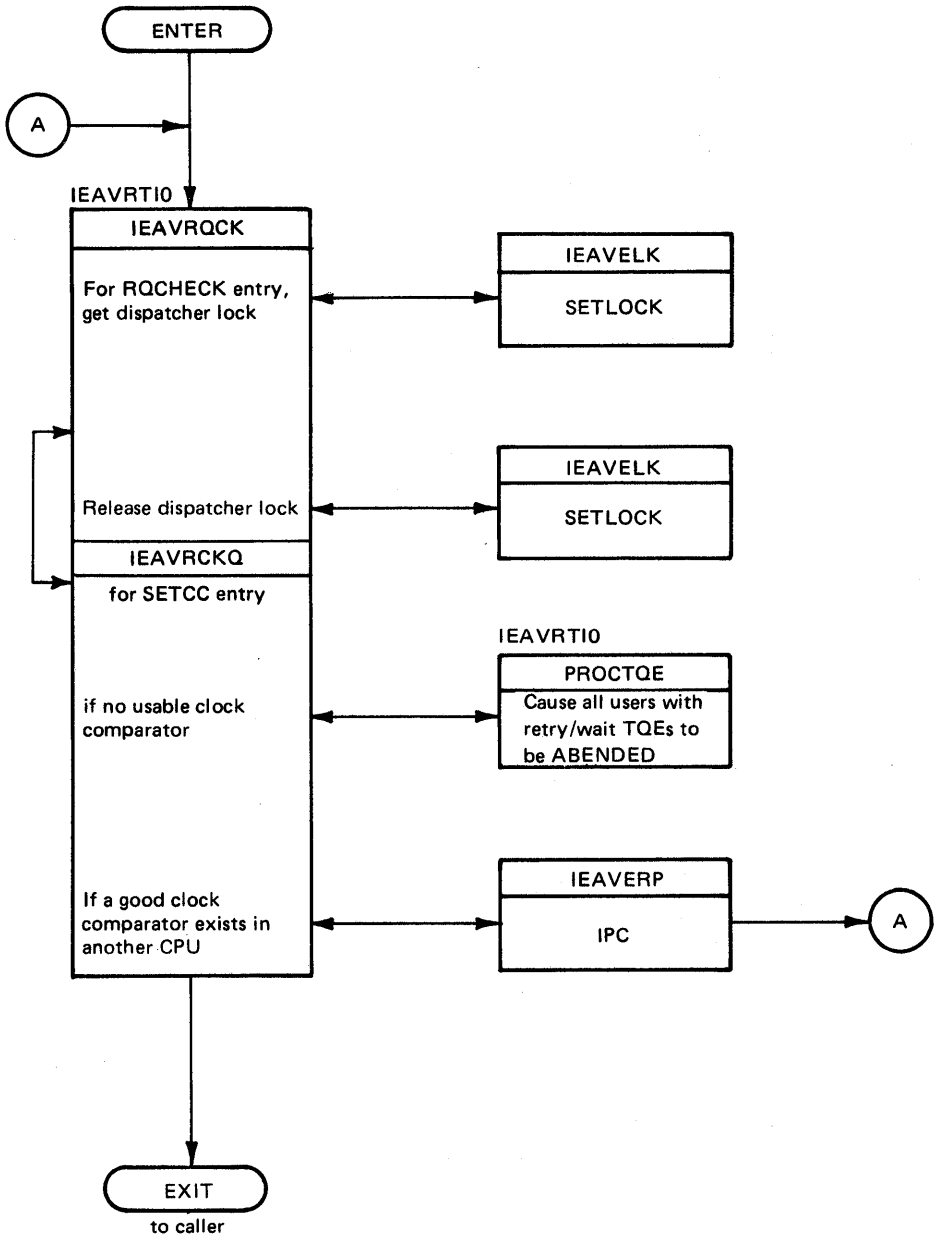
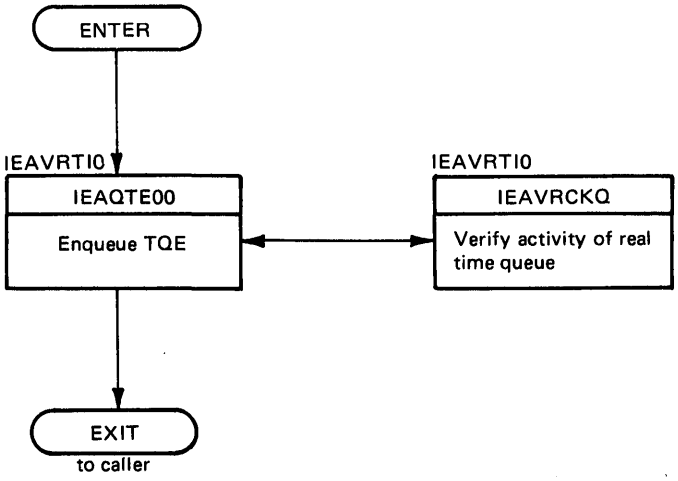


Figure 3-43. Timing Services Module Flow (Part 14 of 23)

TQE Enqueue



TQE Dequeue

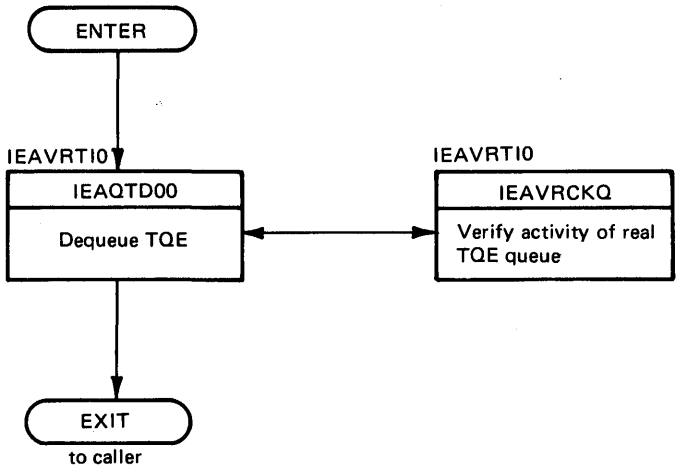


Figure 3-43. Timing Services Module Flow (Part 15 of 23)

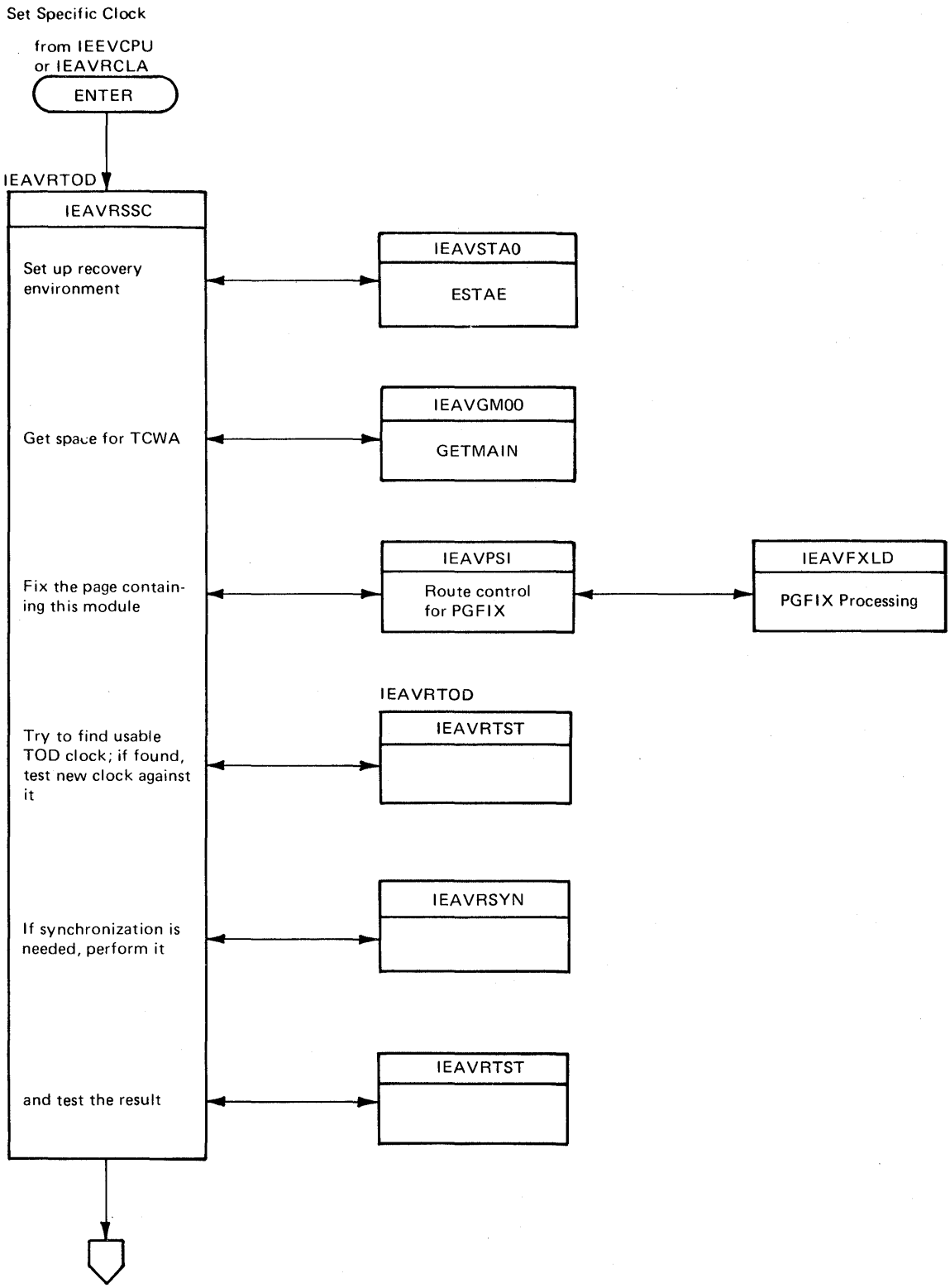


Figure 3-43. Timing Services Module Flow (Part 16 of 23)

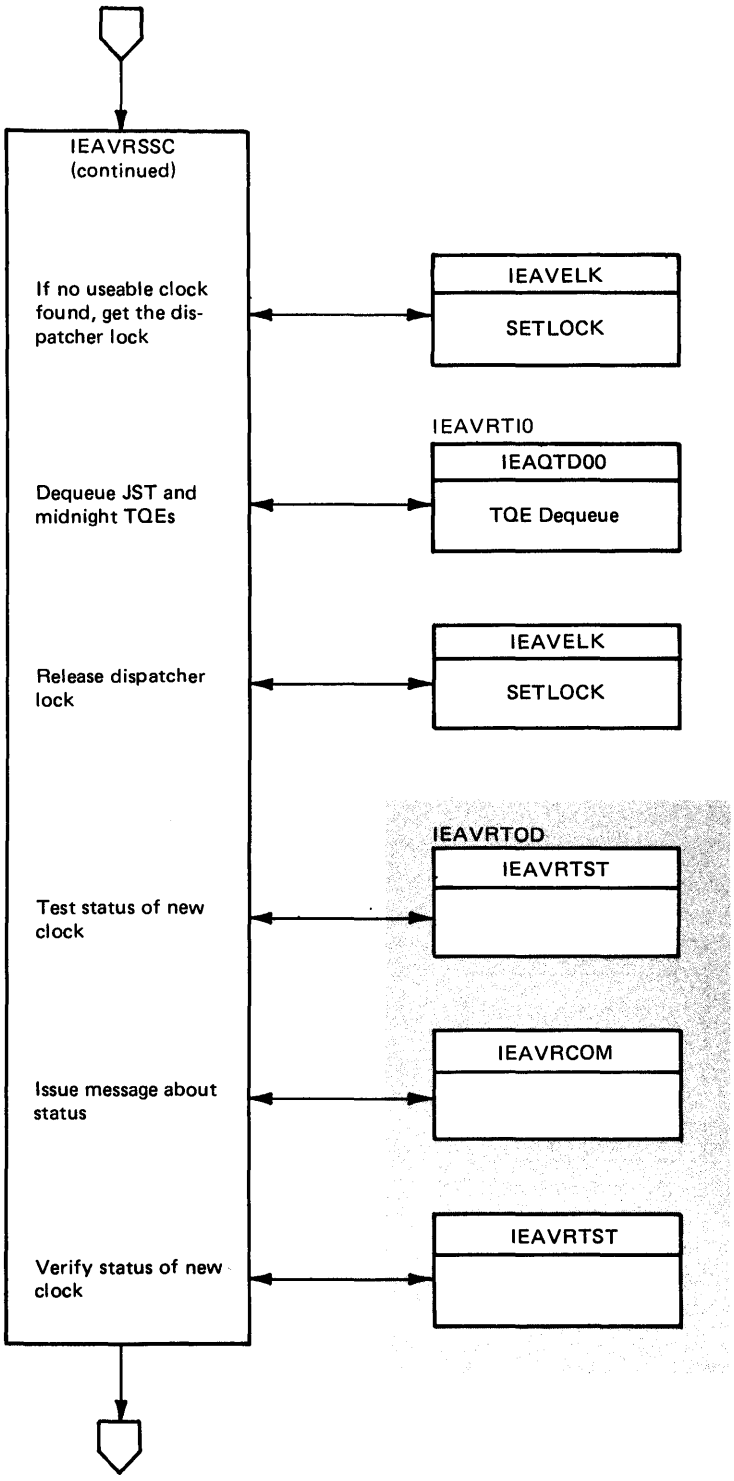


Figure 3-43. Timing Services Module Flow (Part 17 of 23)

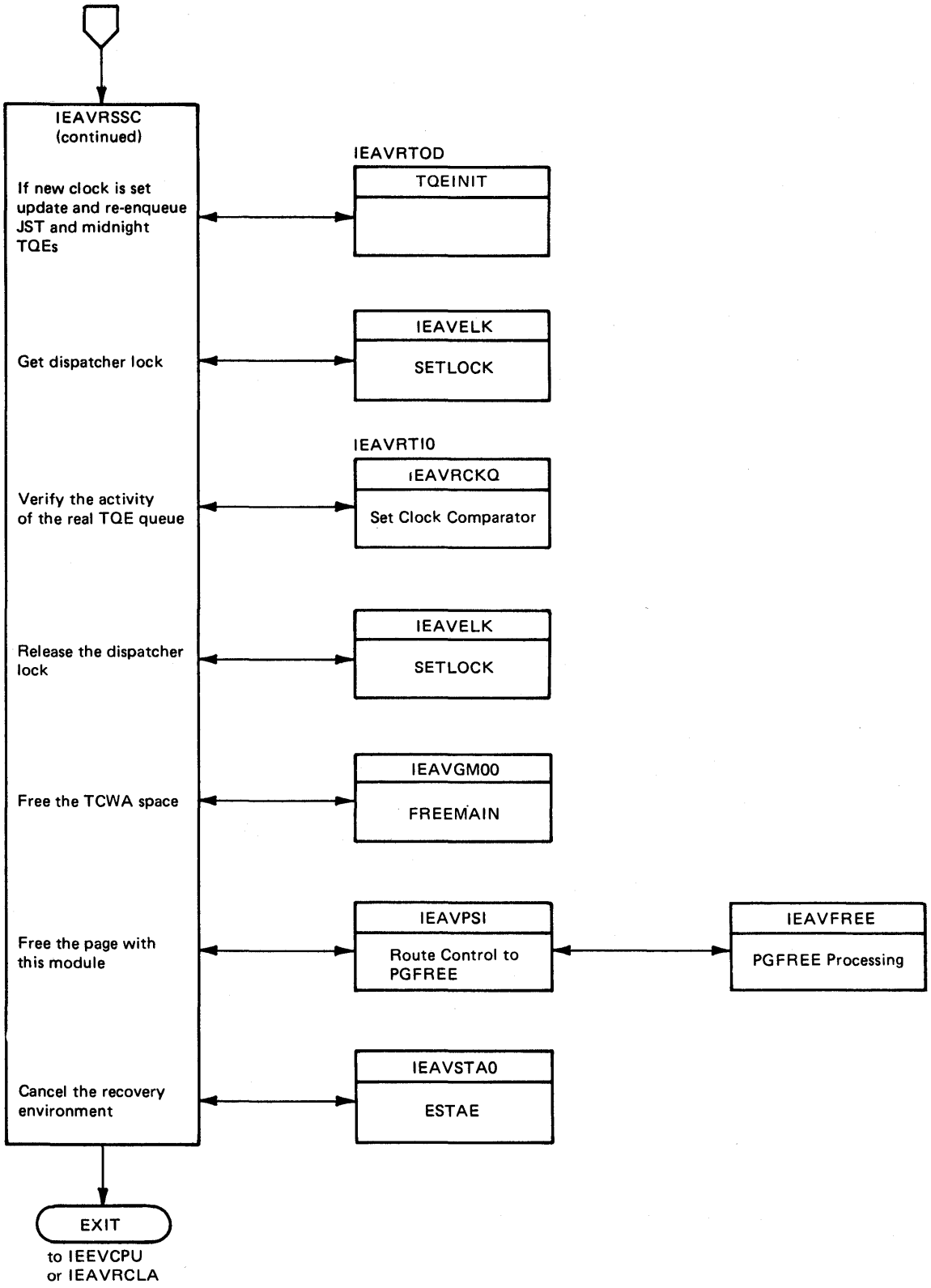


Figure 3-43. Timing Services Module Flow (Part 18 of 23)

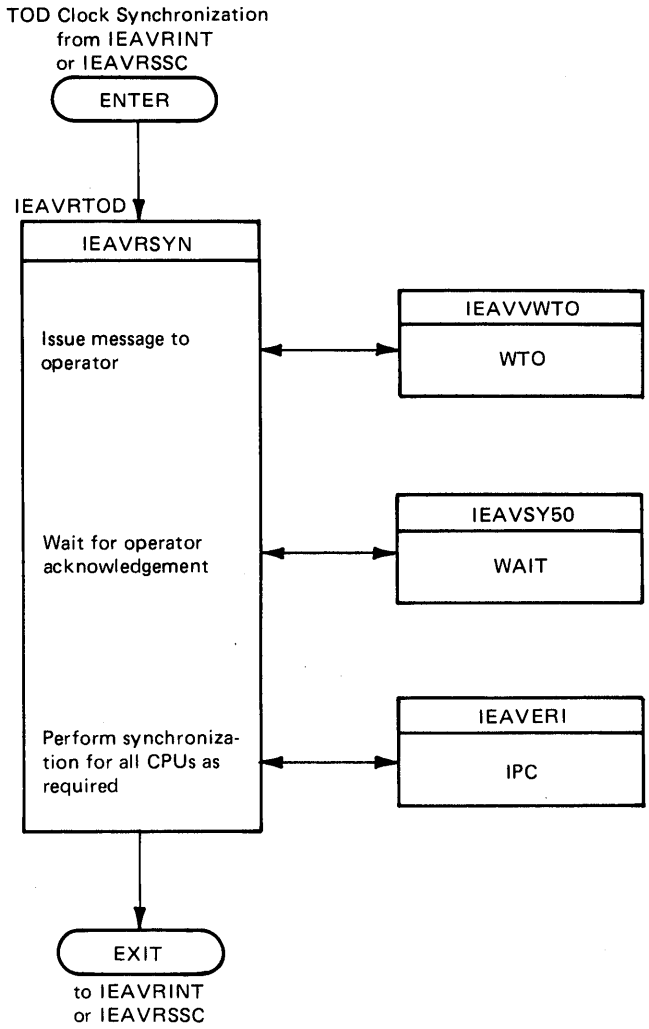


Figure 3-43. Timing Services Module Flow (Part 19 of 23)

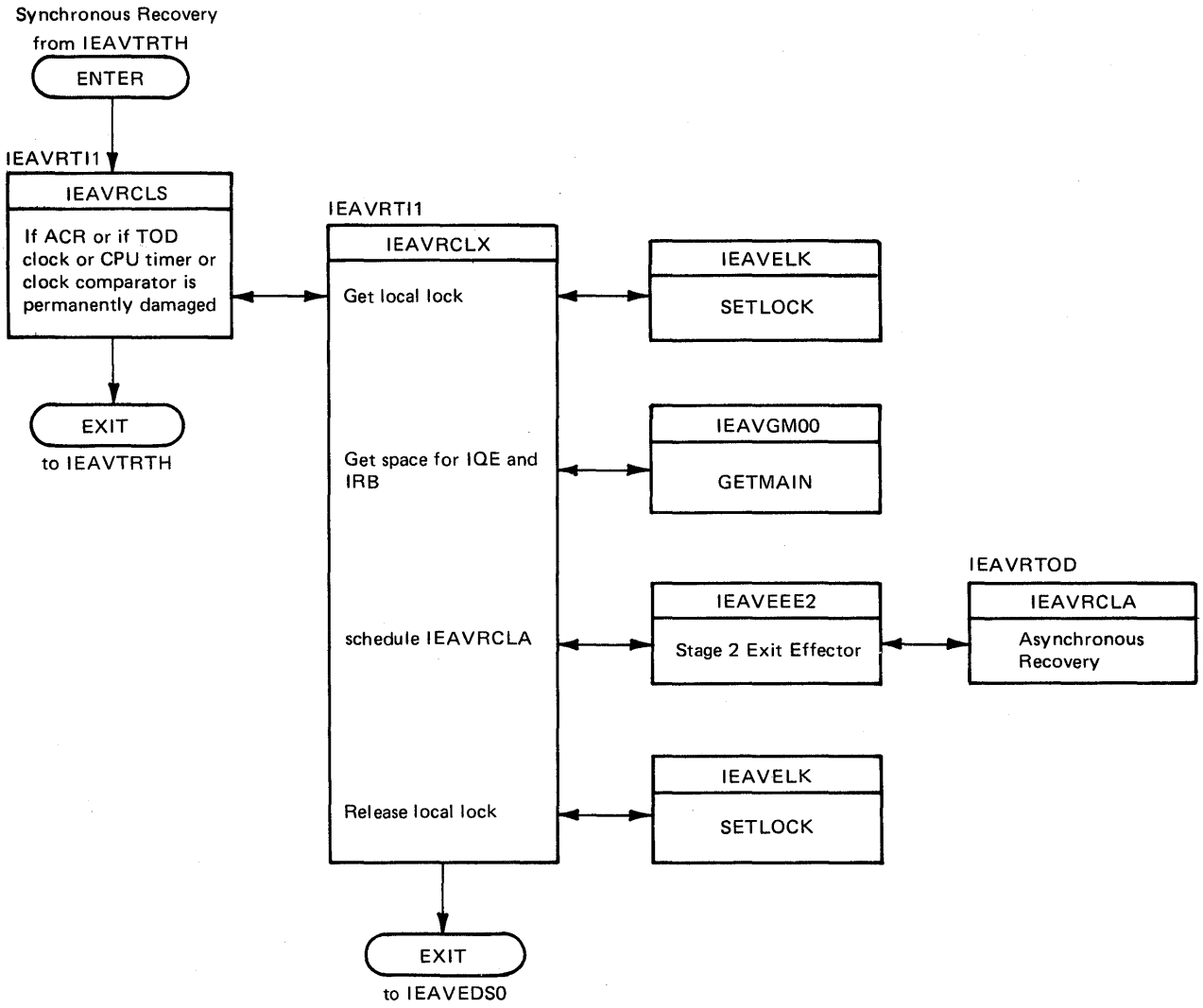


Figure 3-43. Timing Services Module Flow (Part 20 of 23)

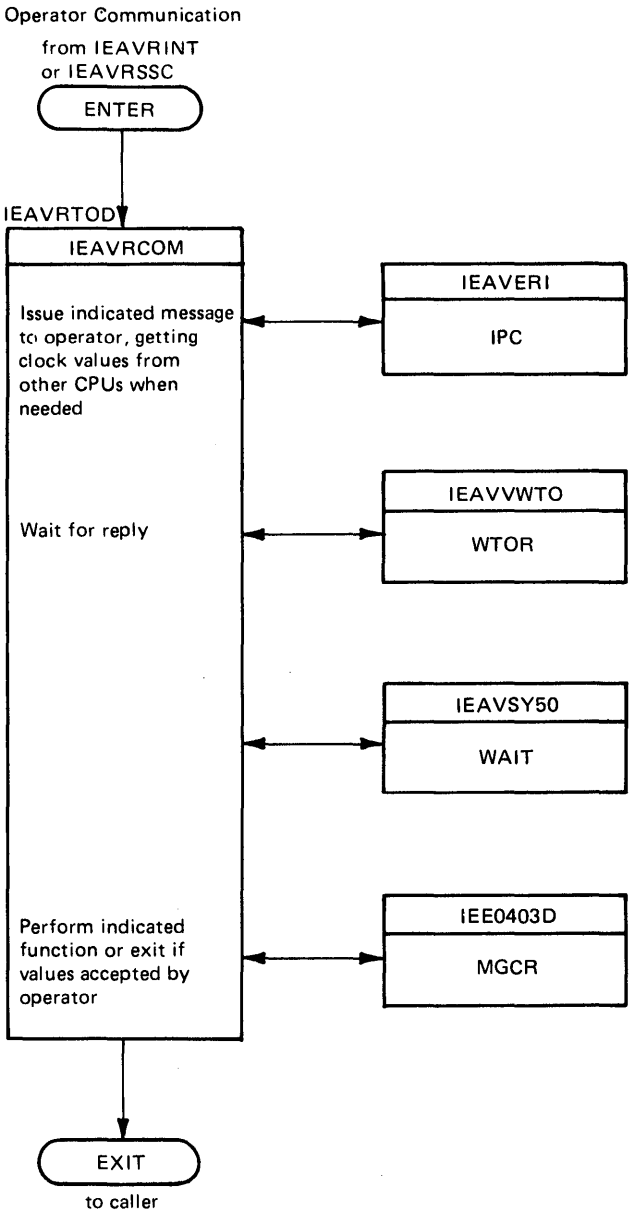


Figure 3-43. Timing Services Module Flow (Part 21 of 23)

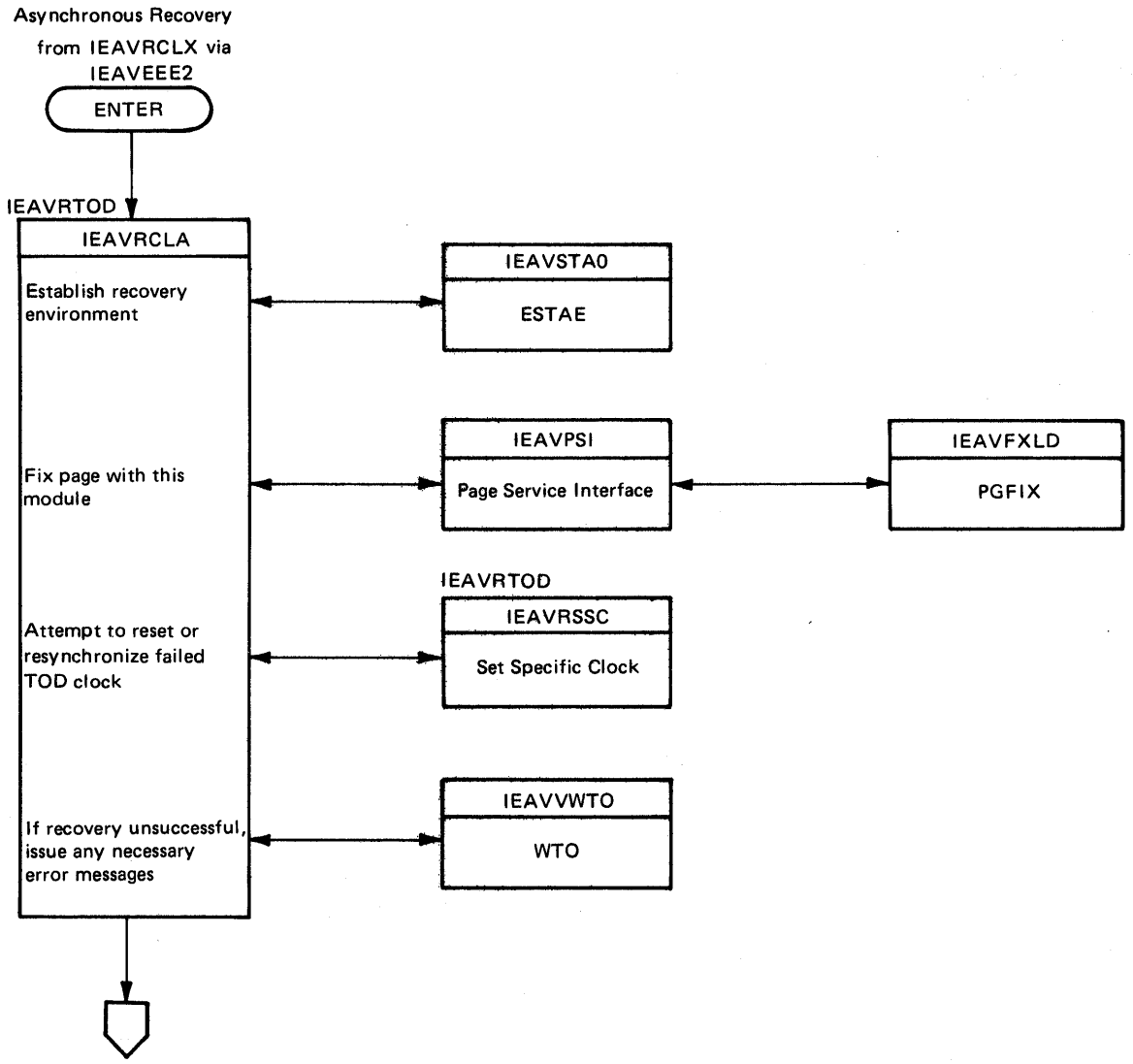


Figure 3-43. Timing Services Module Flow (Part 22 of 23)

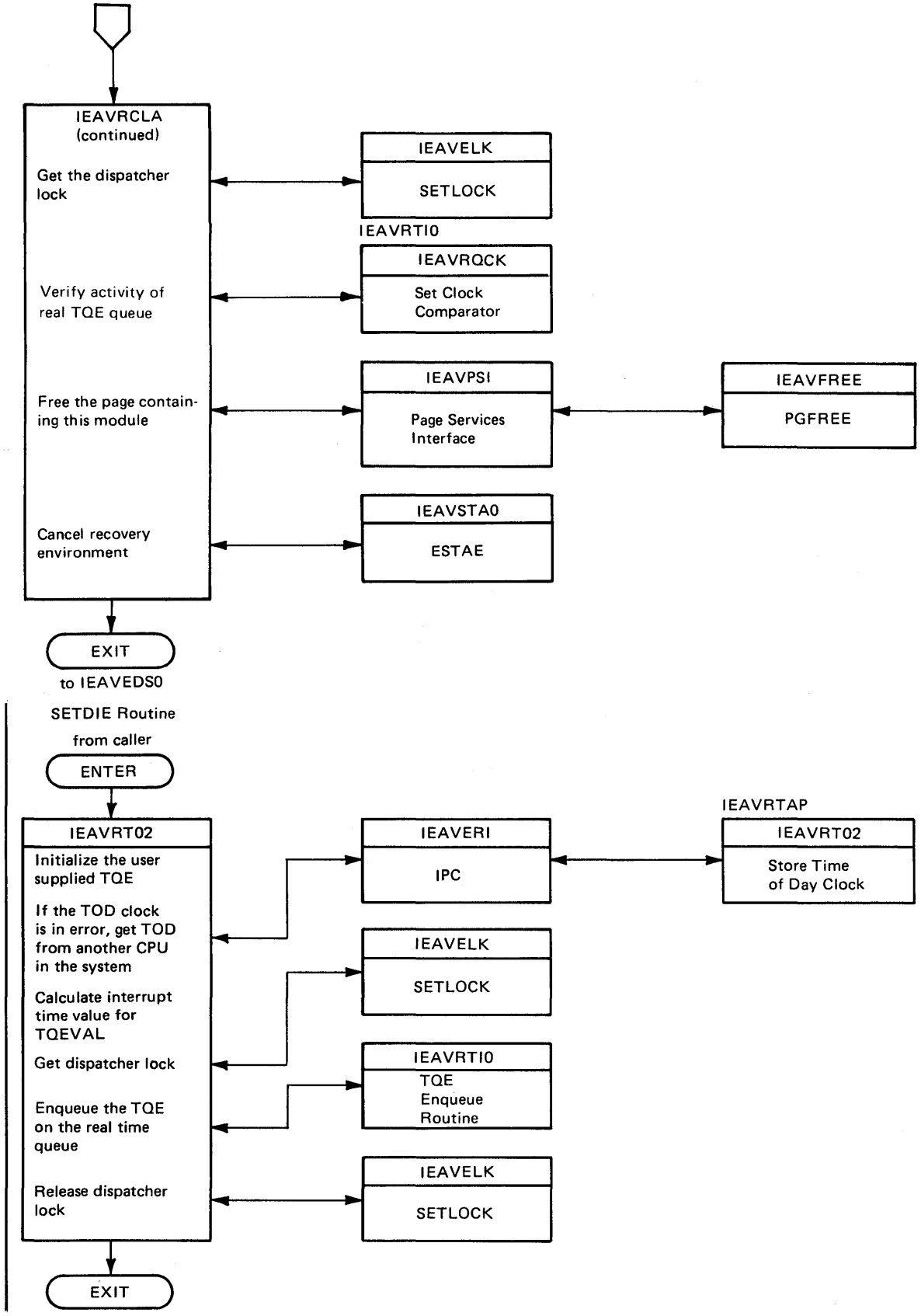


Figure 3-43. Timing Services Module Flow (Part 23 of 23)

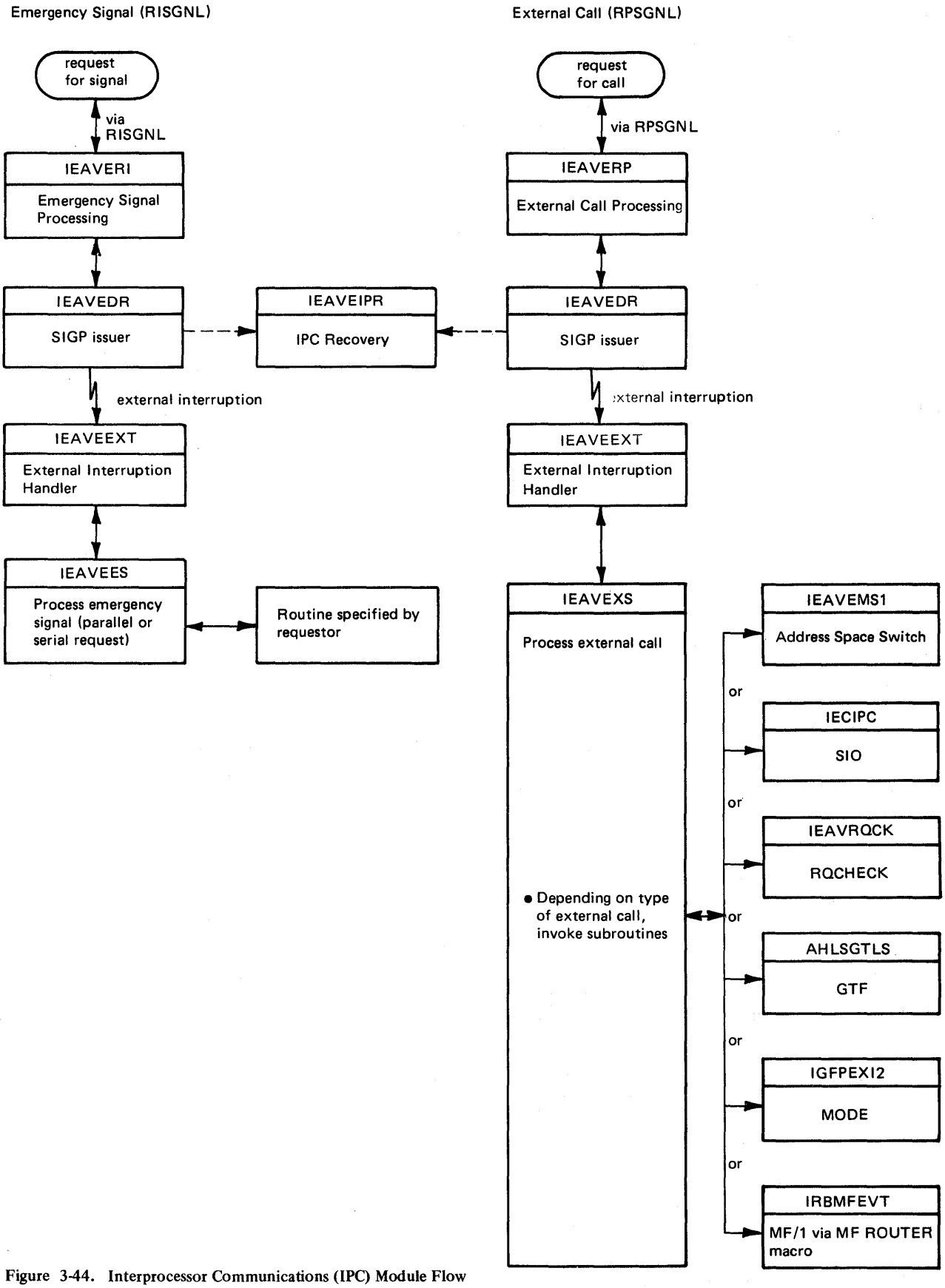


Figure 3-44. Interprocessor Communications (IPC) Module Flow

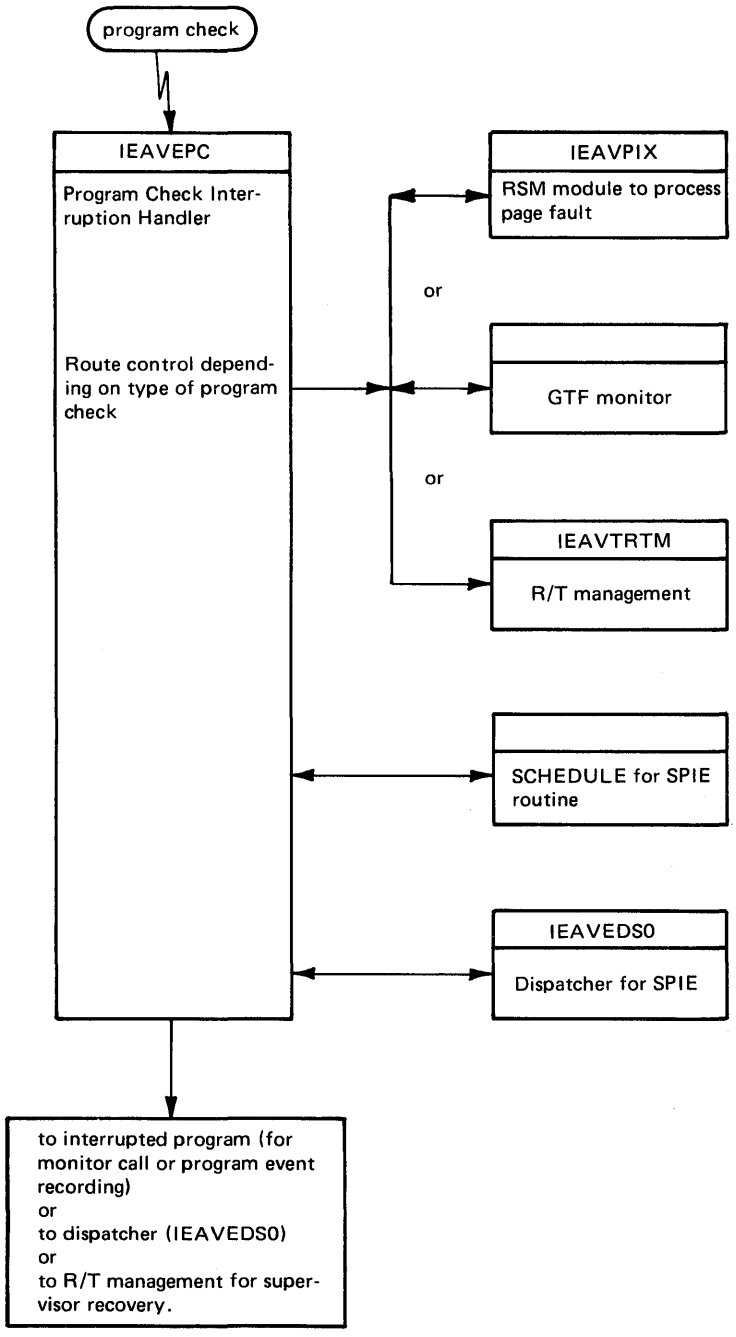


Figure 3-45. Interruption Handlers Module Flow (Part 1 of 4)

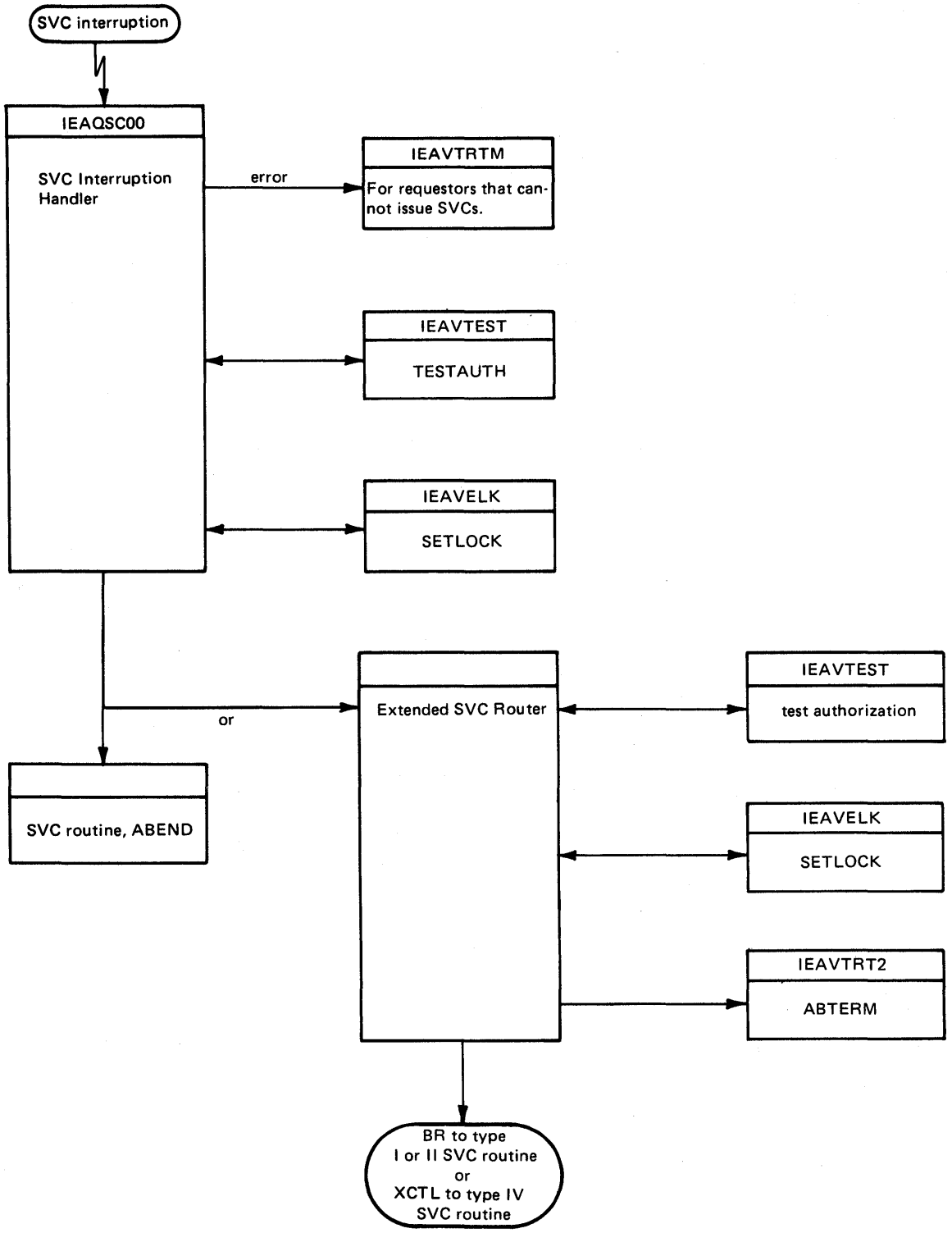


Figure 3-45. Interruption Handlers Module Flow (Part 2 of 4)

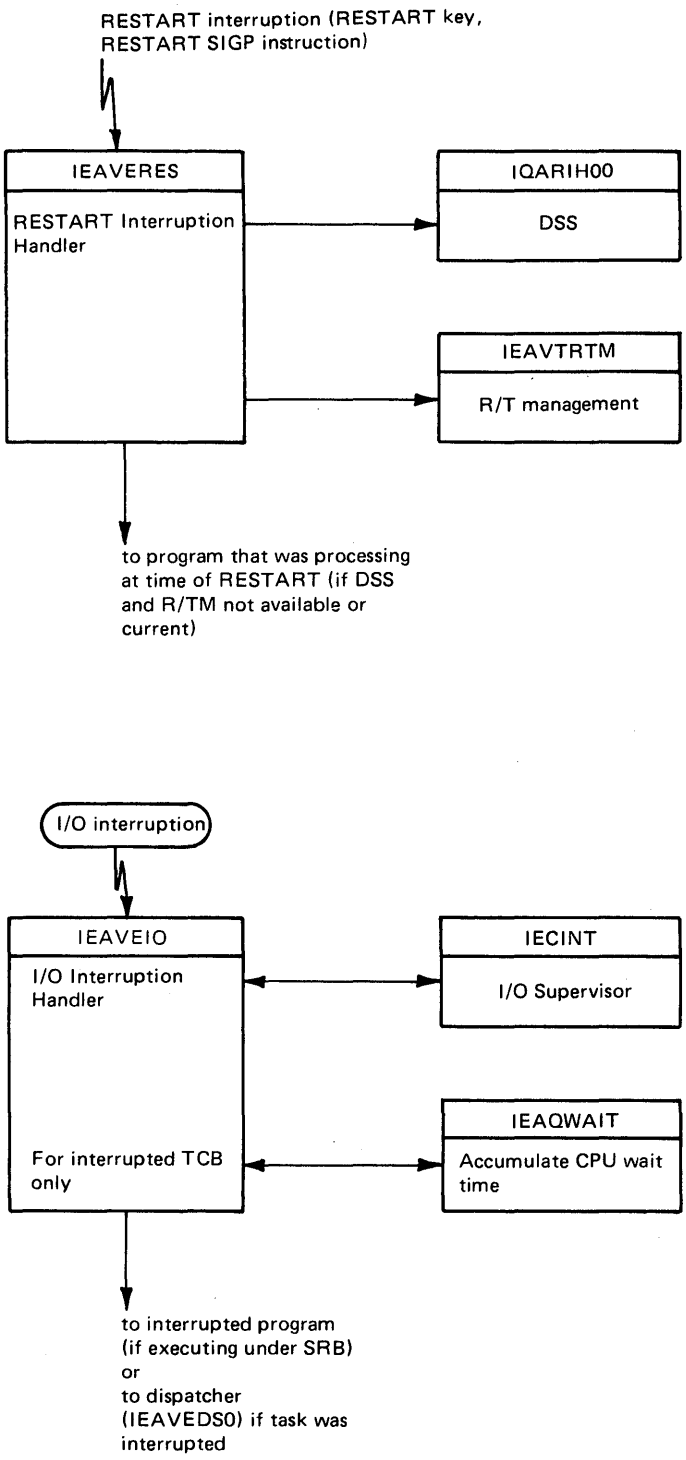


Figure 3-45. Interruption Handlers Module Flow (Part 3 of 4)

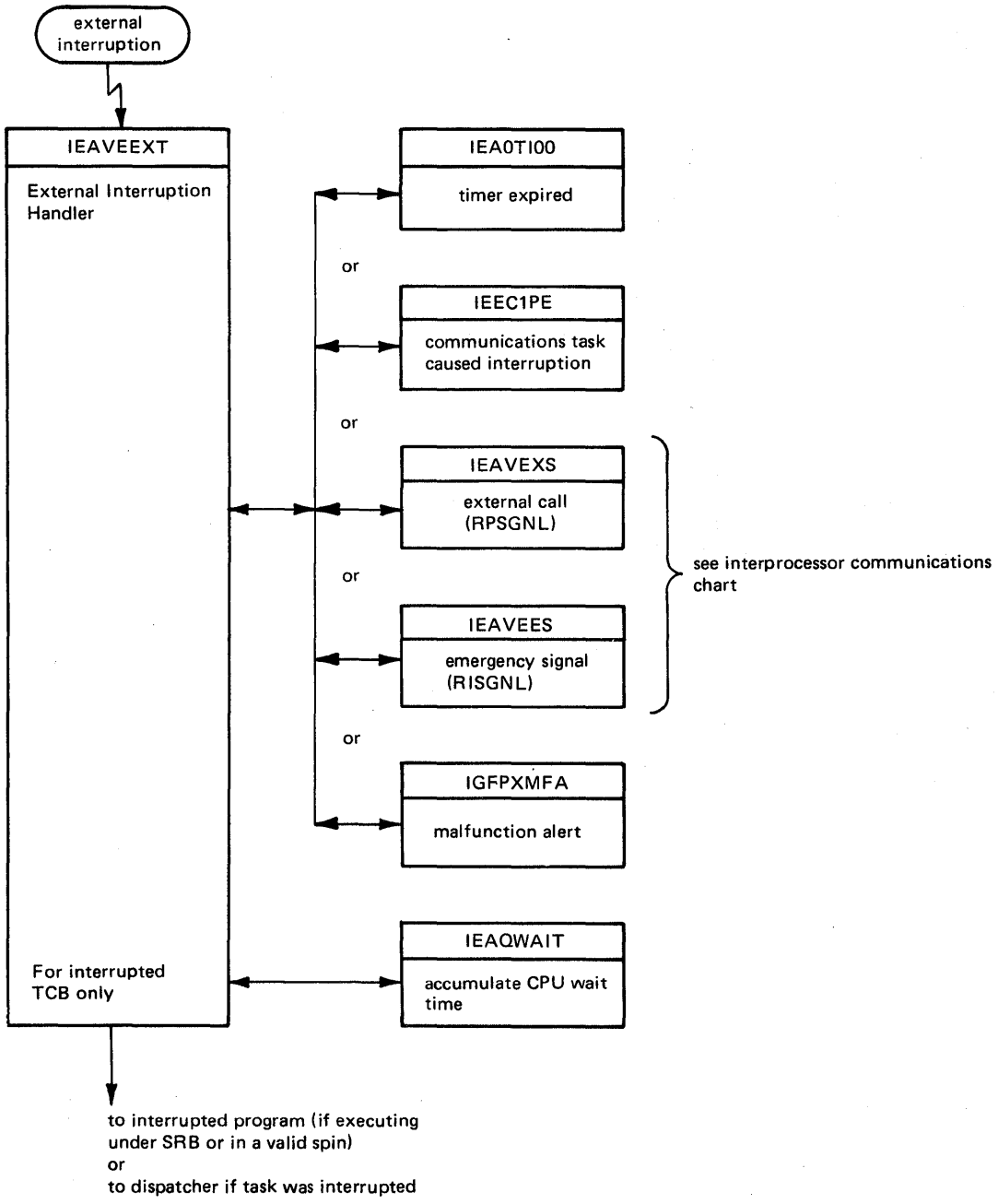


Figure 3-45. Interruption Handlers Module Flow (Part 4 of 4)

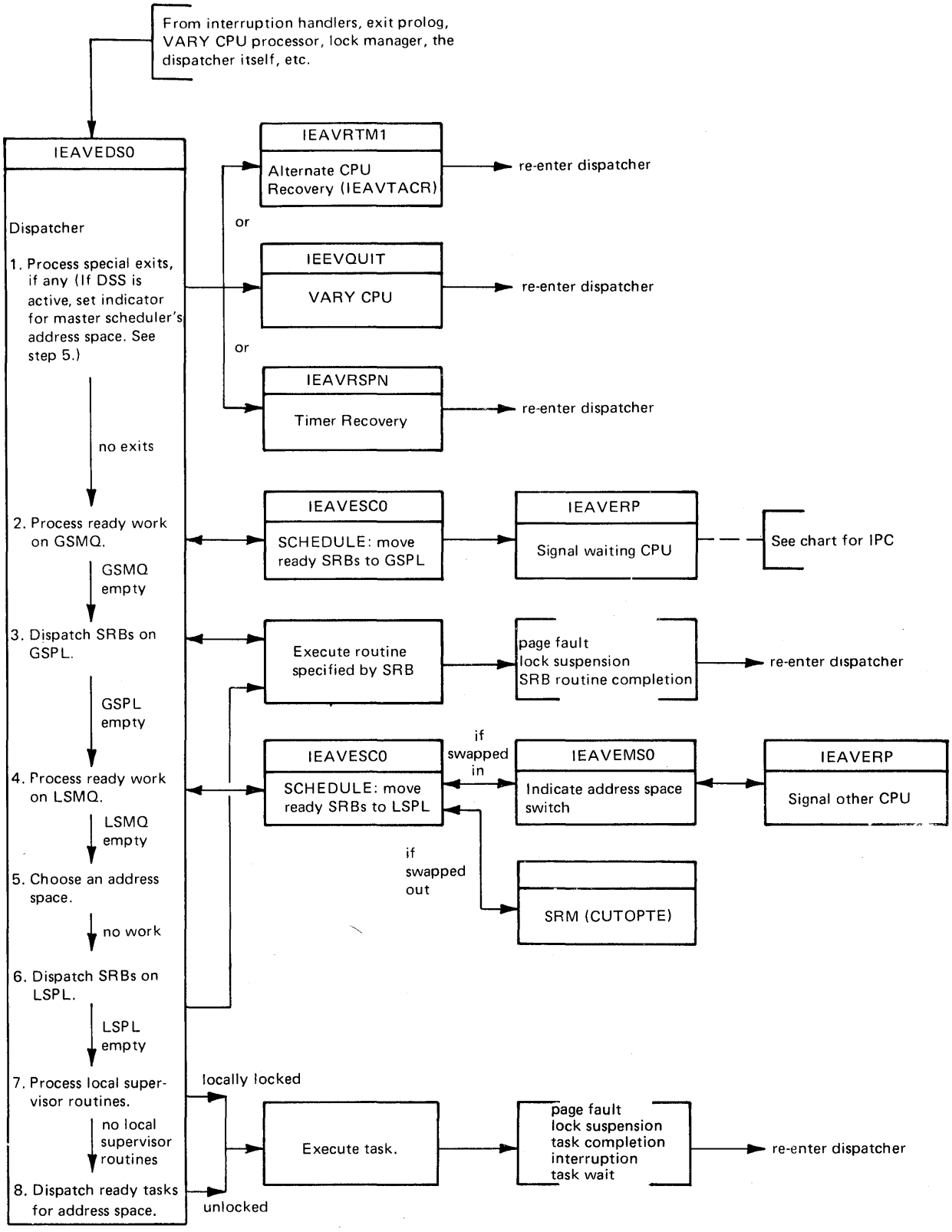


Figure 3-46. Dispatcher and SCHEDULE Module Flow

Exit Effectors (for asynchronous exits)

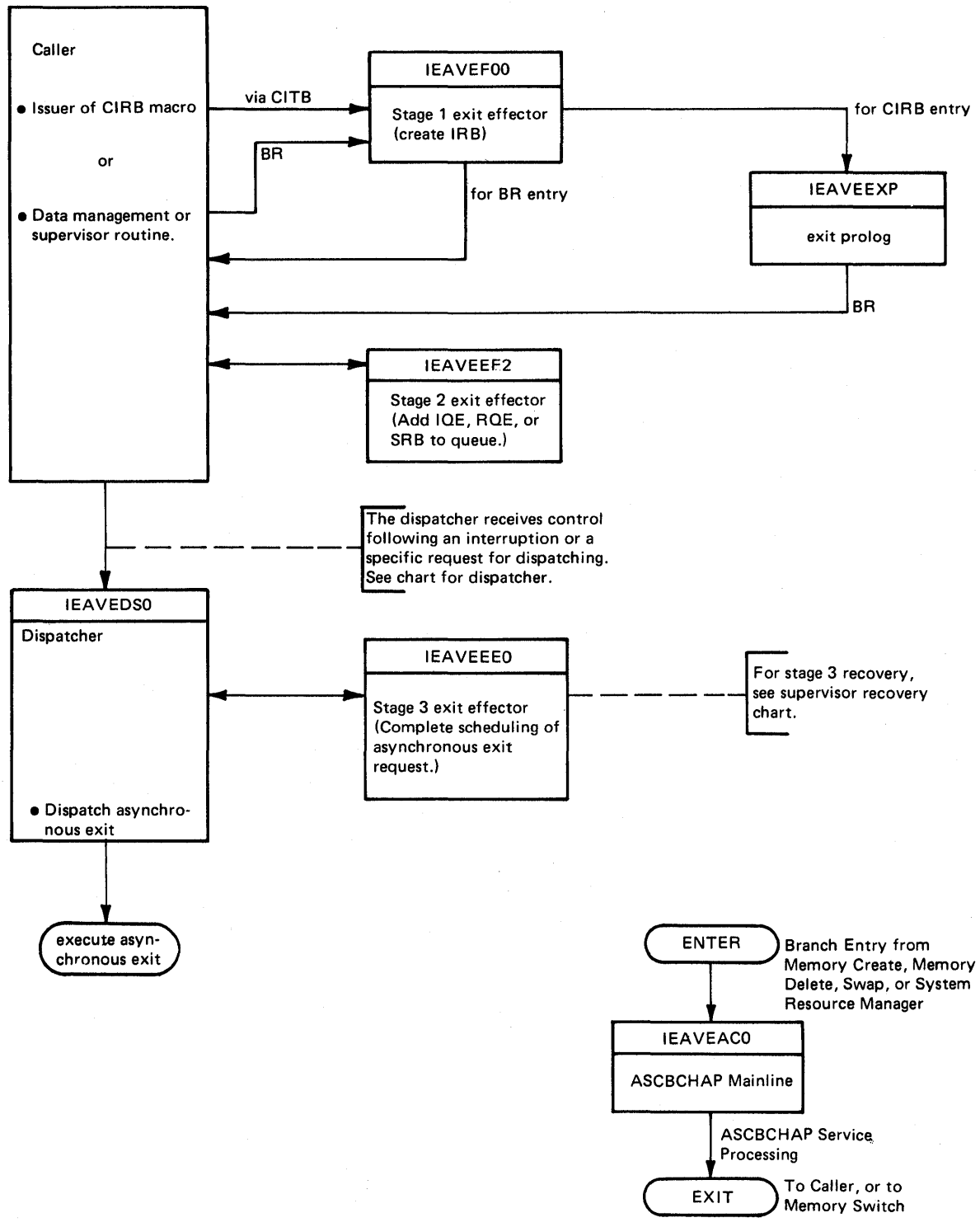


Figure 3-47. Supervisor Routines Module Flow (Part 1 of 2)

SETLOCK

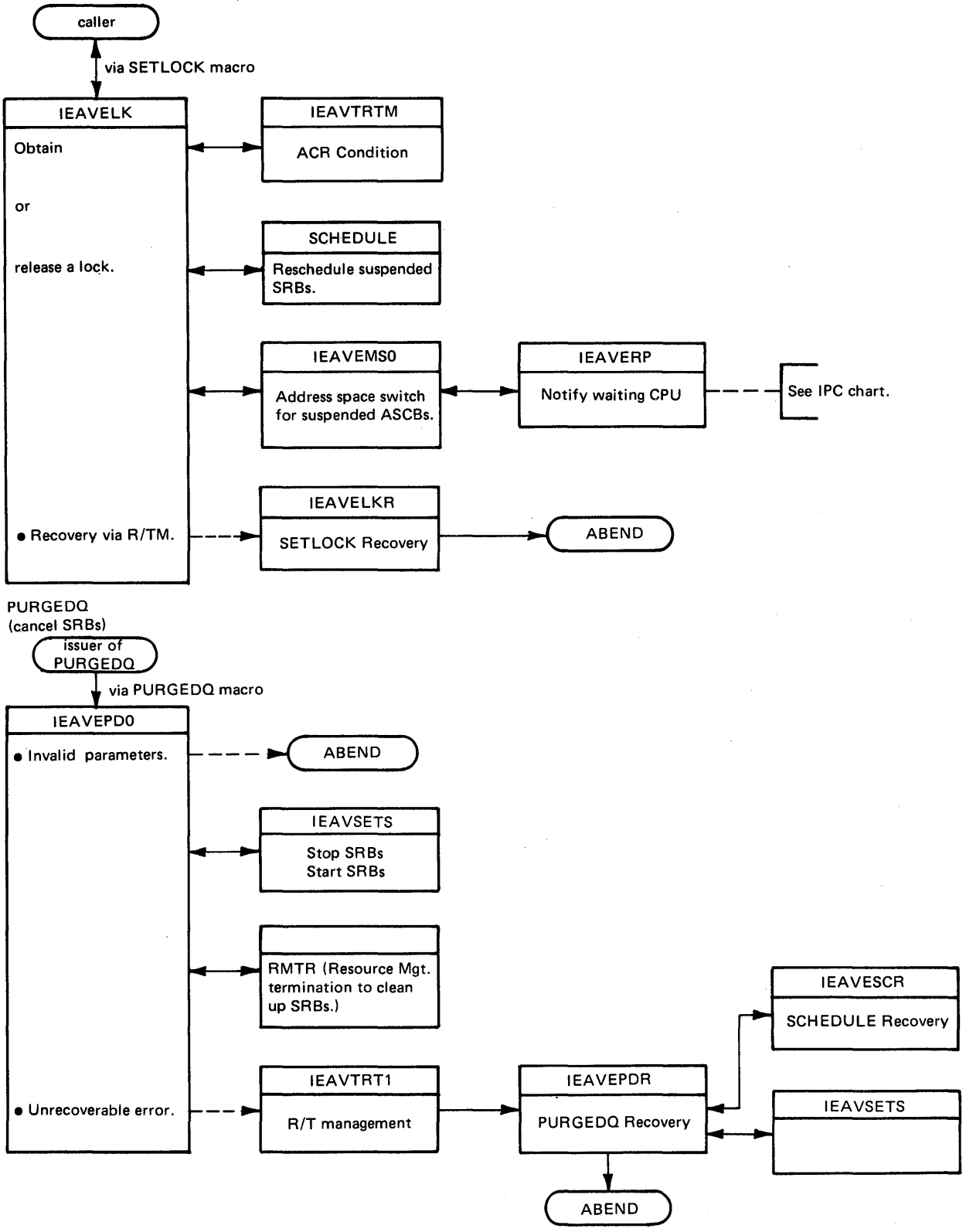


Figure 3-47. Supervisor Routines Module Flow (Part 2 of 2)

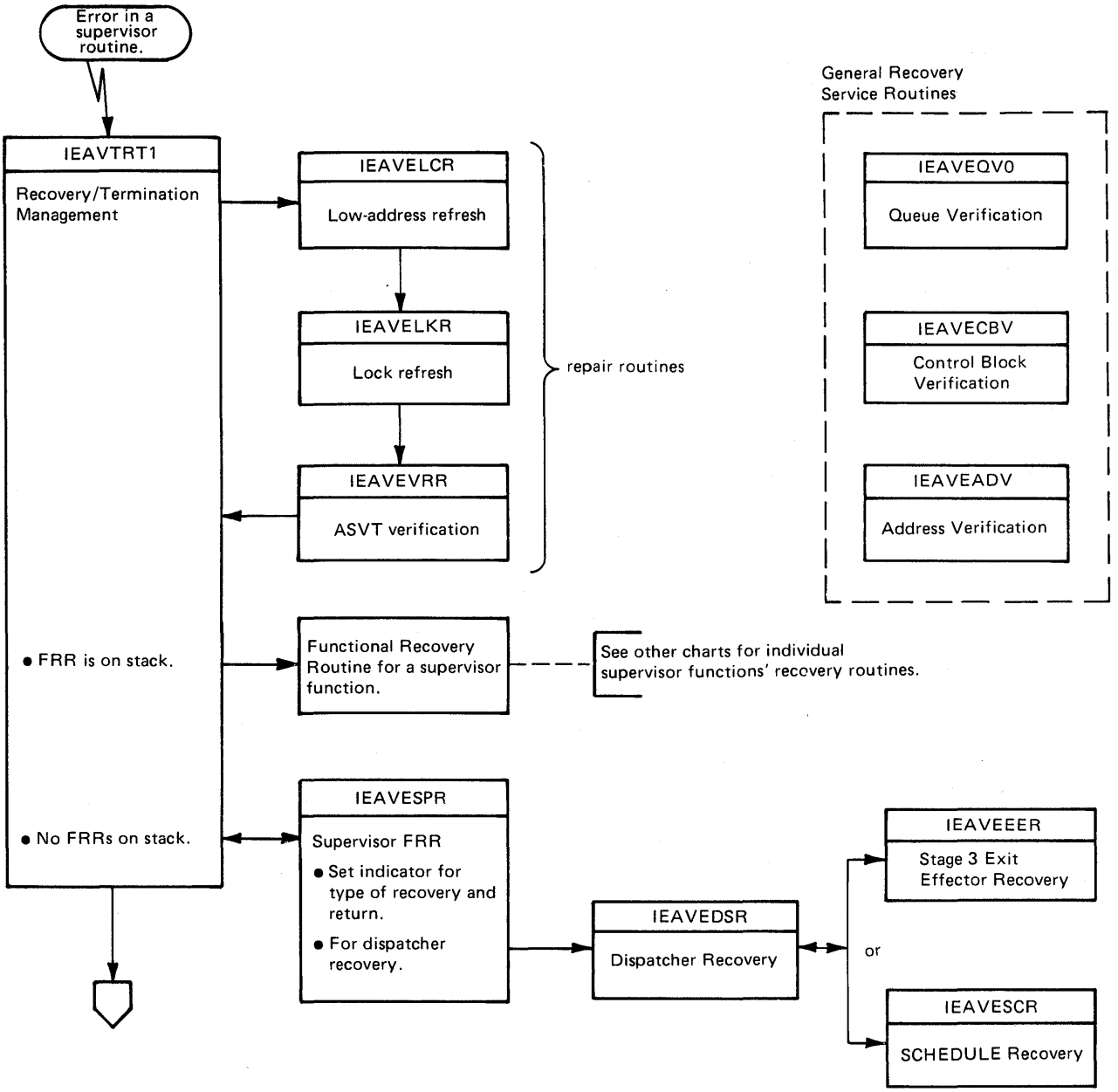


Figure 3-48. Supervisor Control Recovery Module Flow (Part 1 of 2)

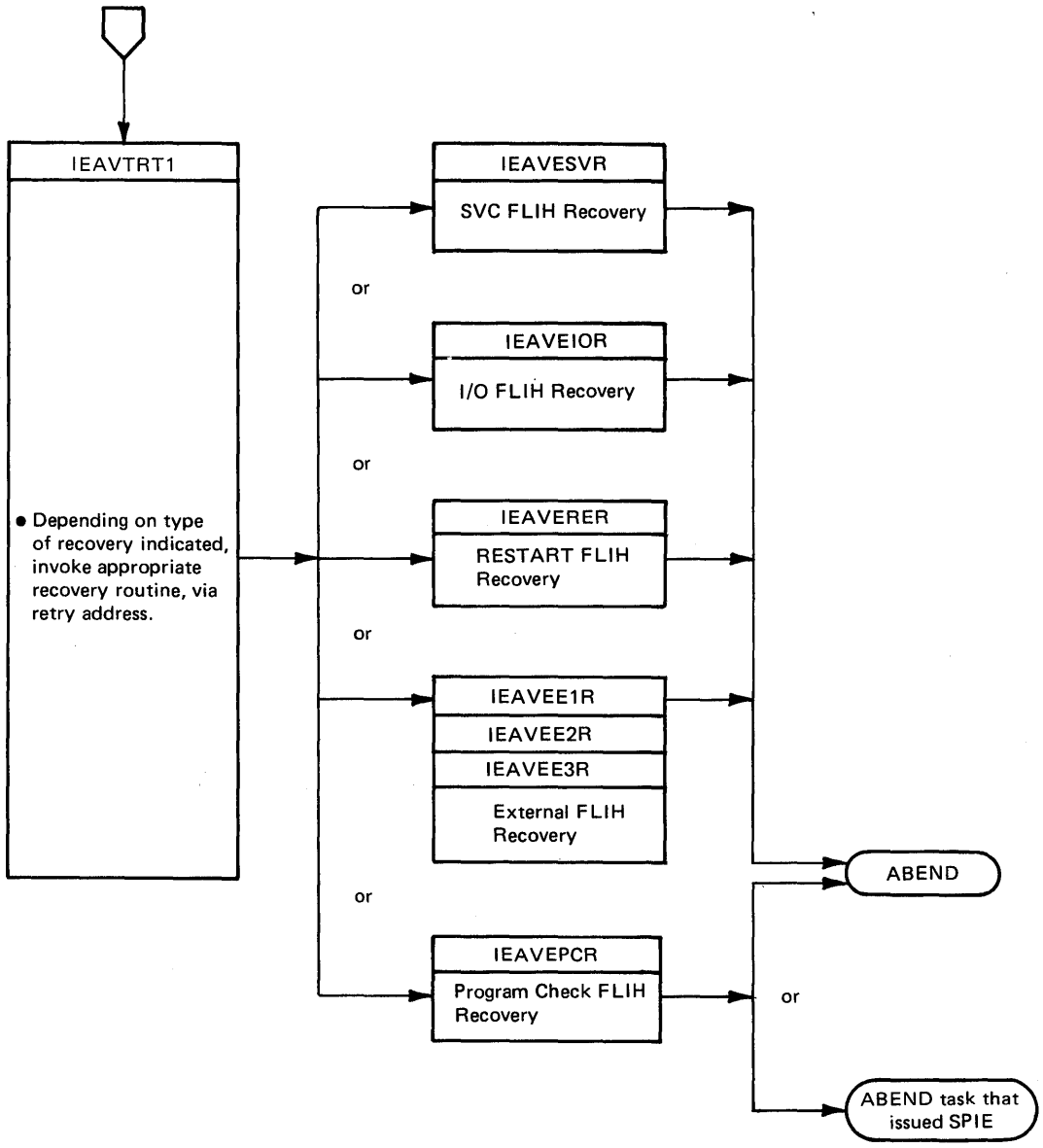


Figure 3-48. Supervisor Control Recovery Module Flow (Part 2 of 2)

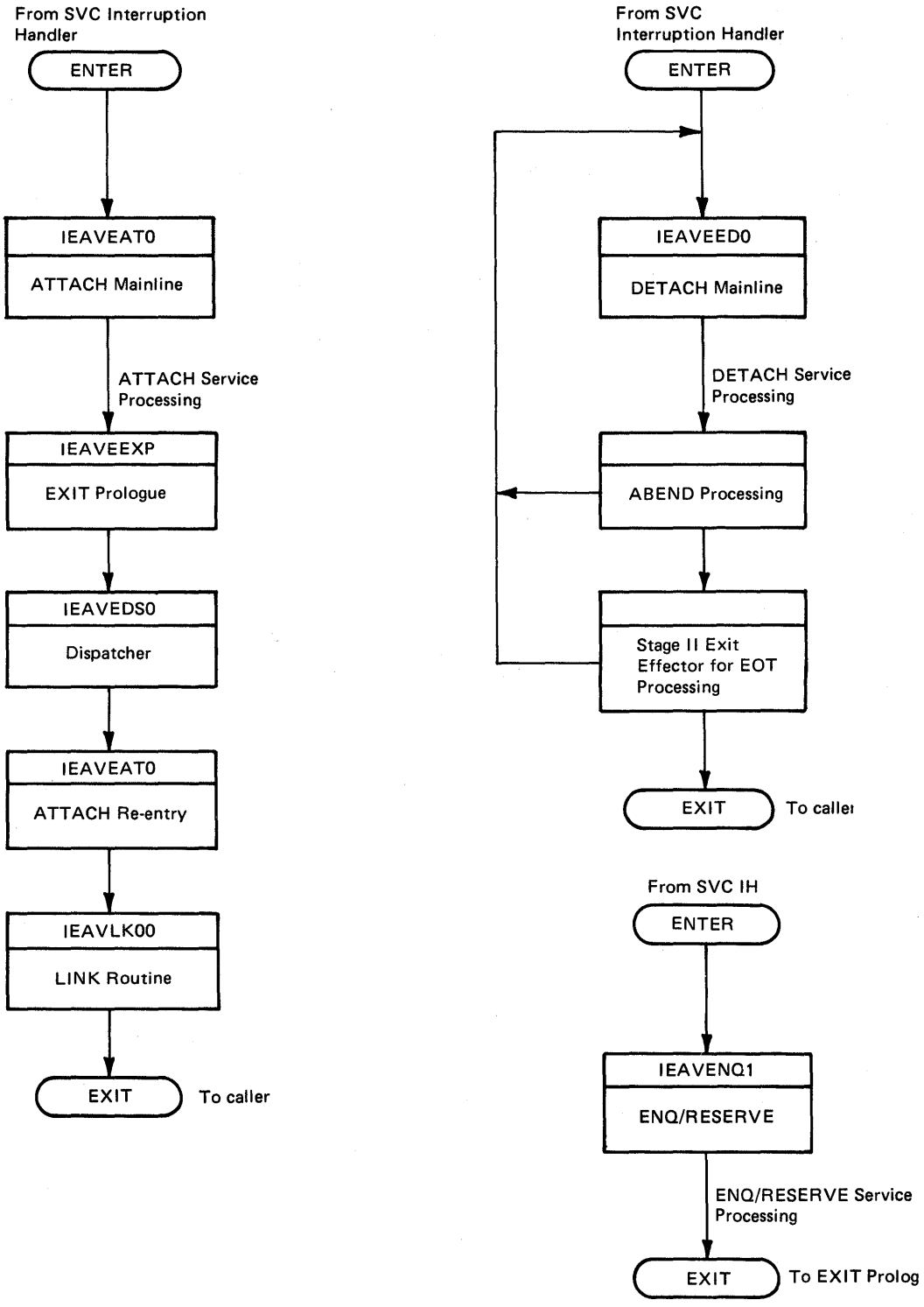
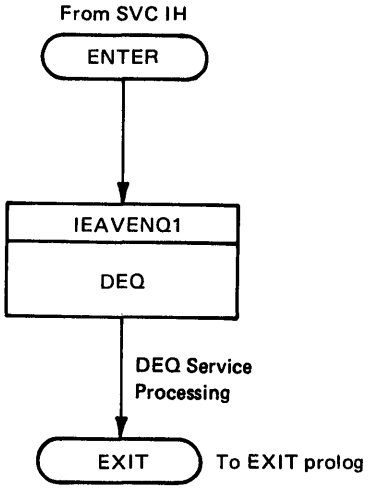
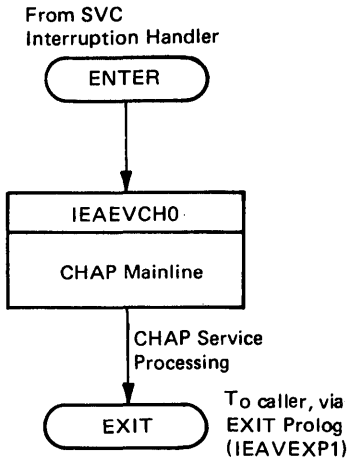
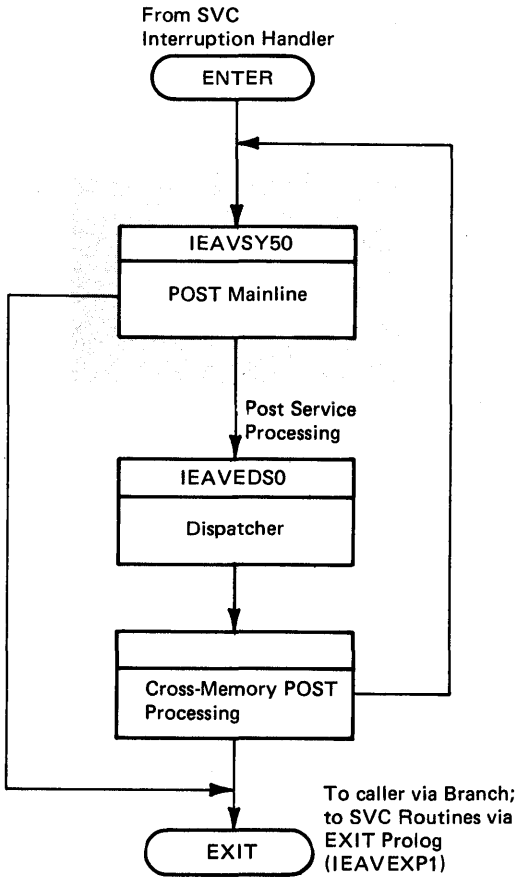


Figure 3-49. ATTACH, DETACH and ENQ/RESERVE Module Flow







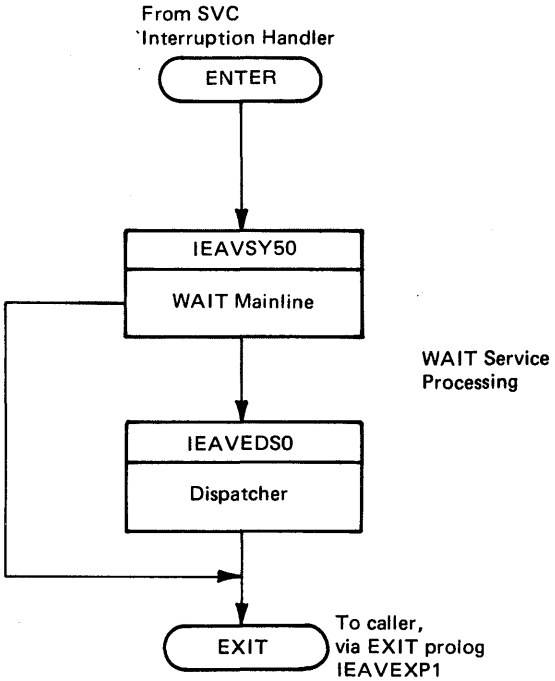


Figure 3-50. DEQ, CHAP, WAIT, and POST Module Flow

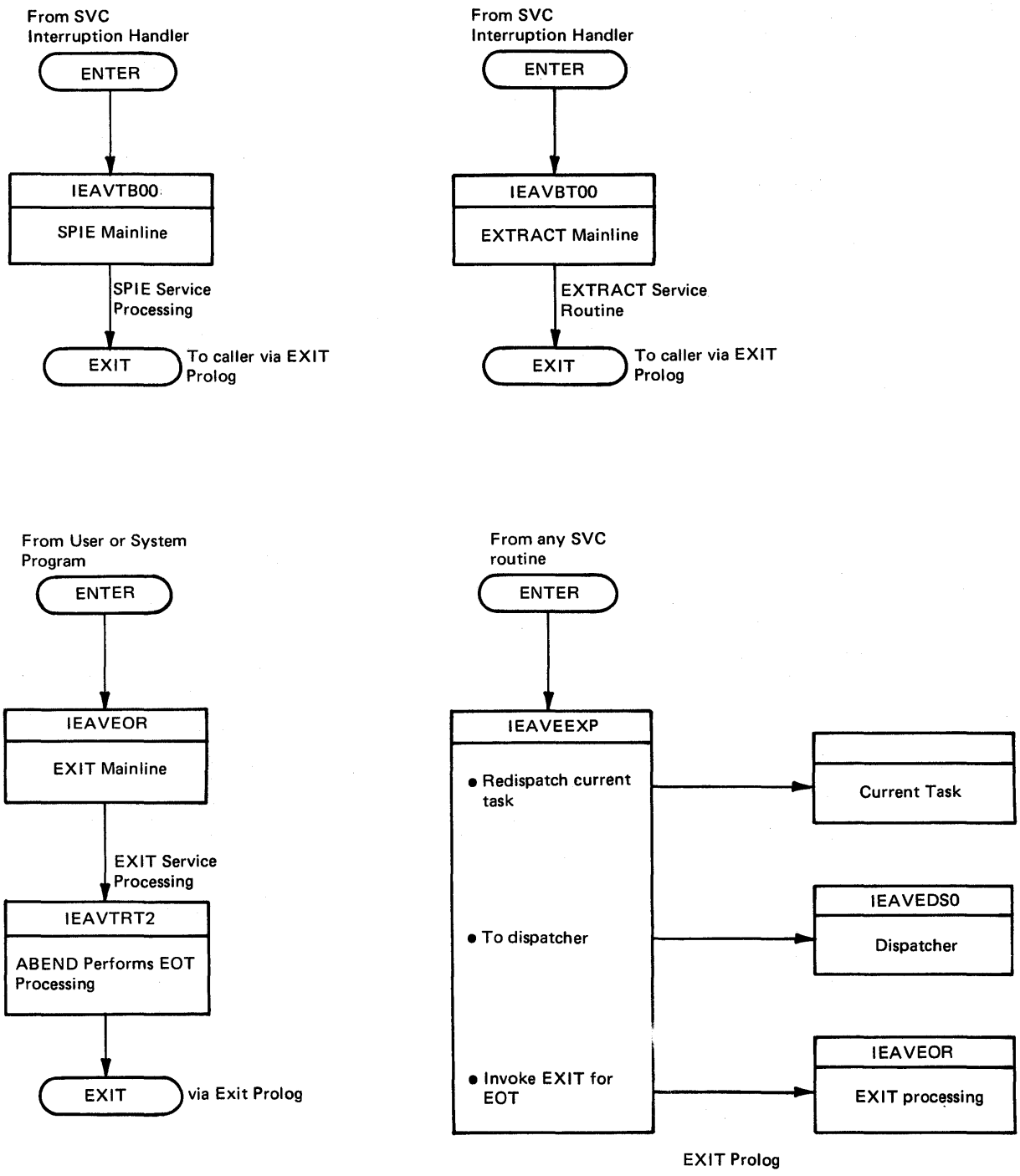


Figure 3-51. SPIE, EXTRACT, EXIT, and Exit Prolog Module Flow

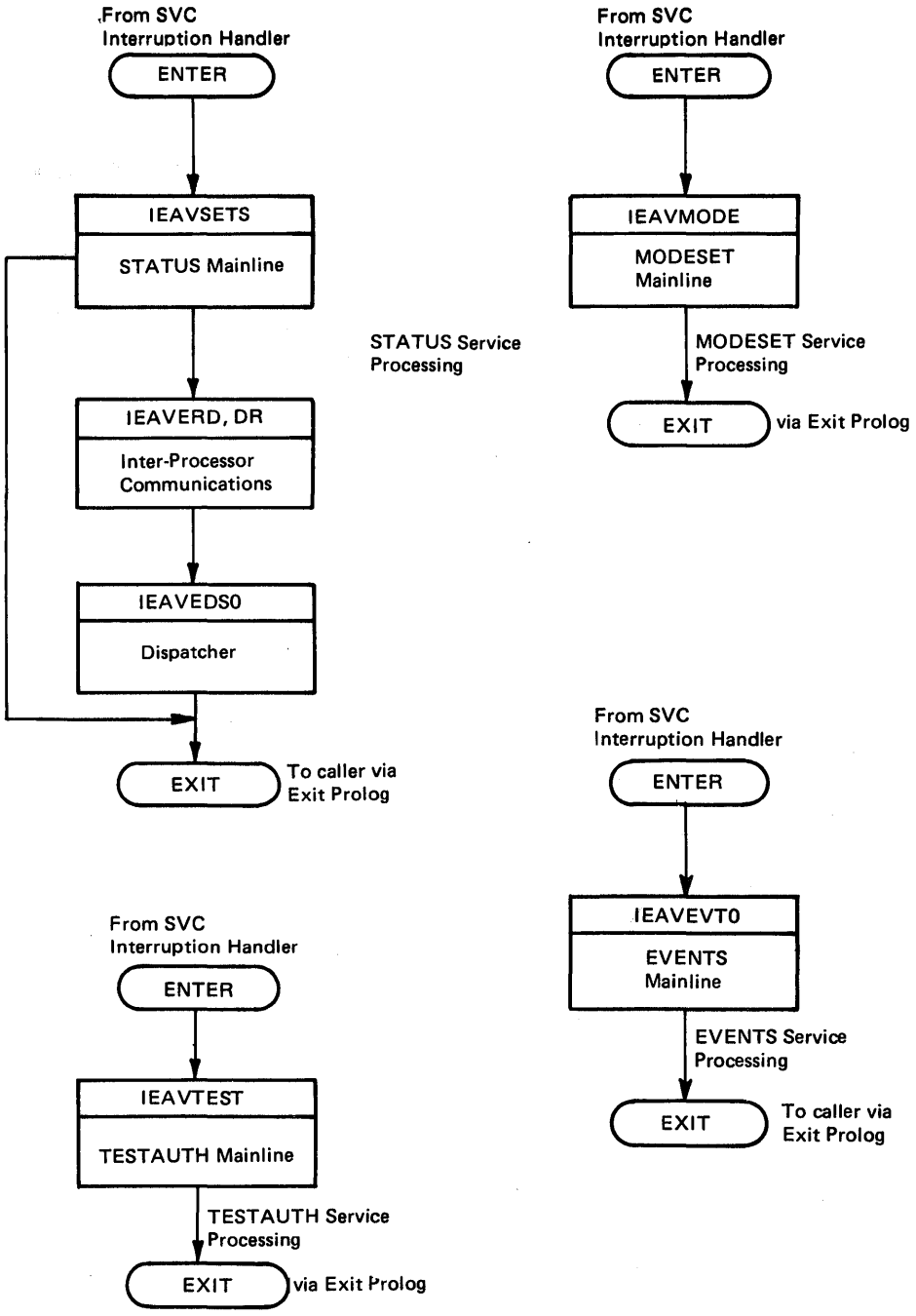


Figure 3-52. STATUS, MODESET, EVENTS, and TESTAUTH Module Flow

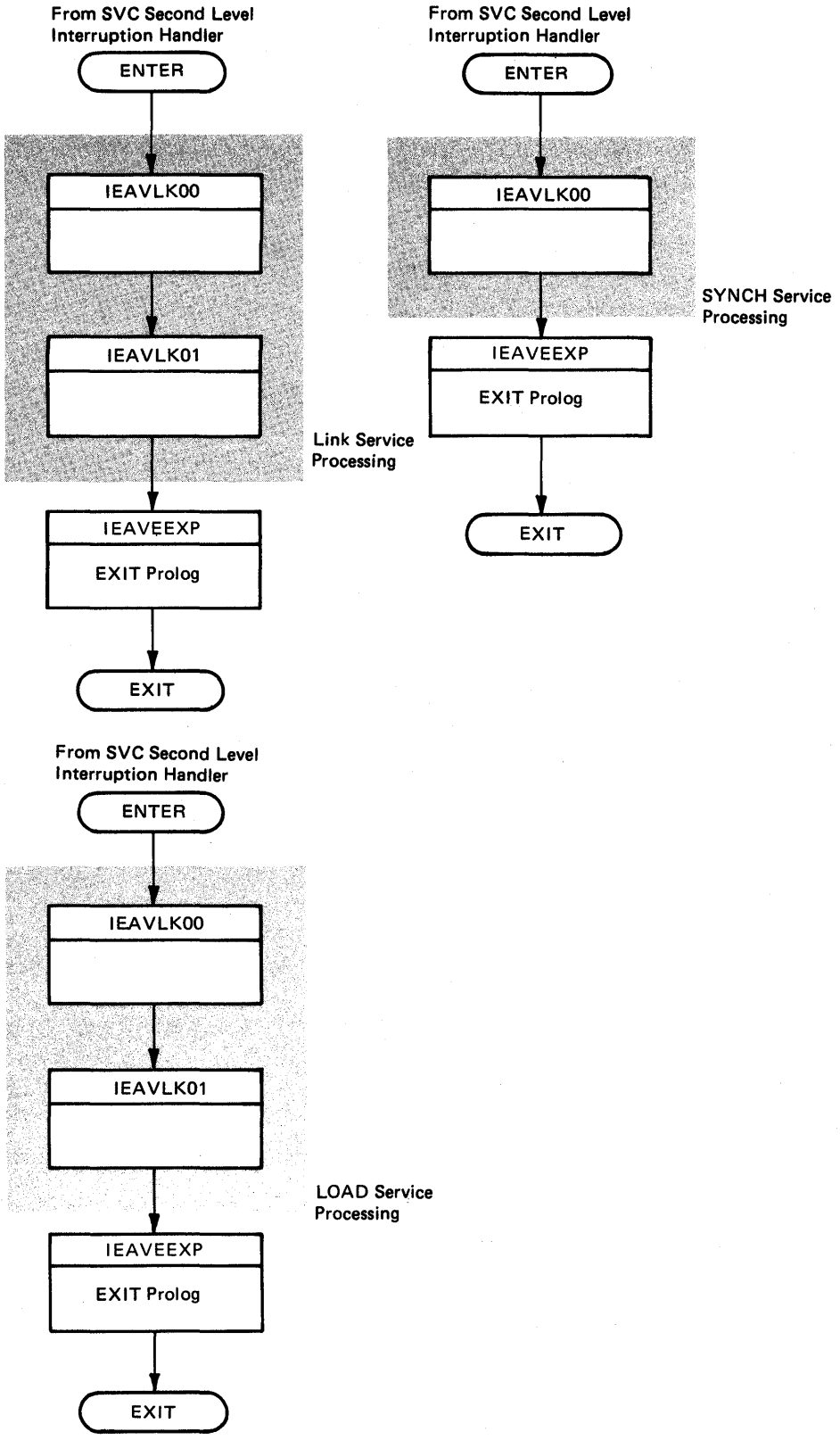


Figure 3-53. LINK, SYNCH, LOAD, Program FETCH/BLDL Interface Module Flow

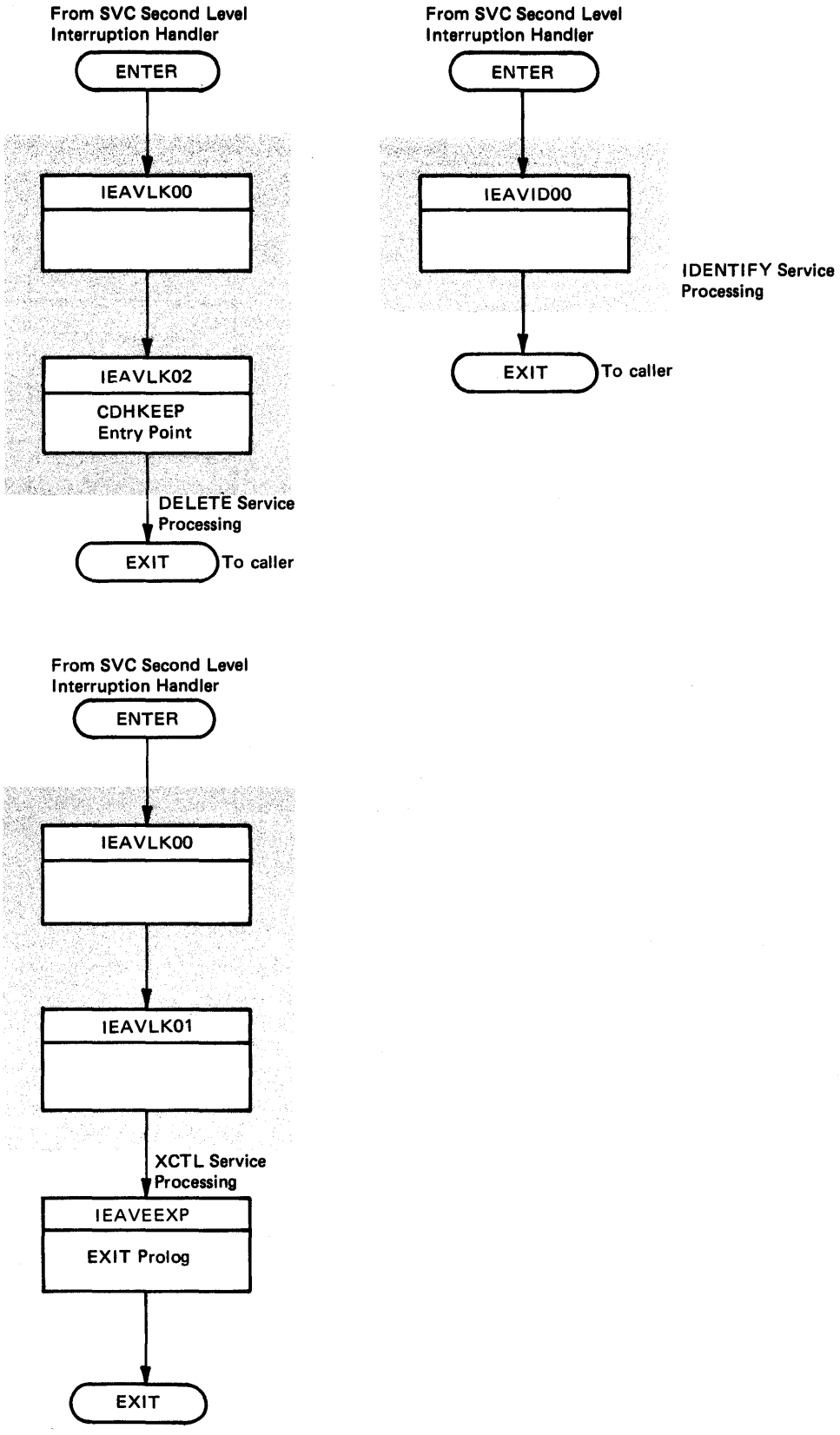
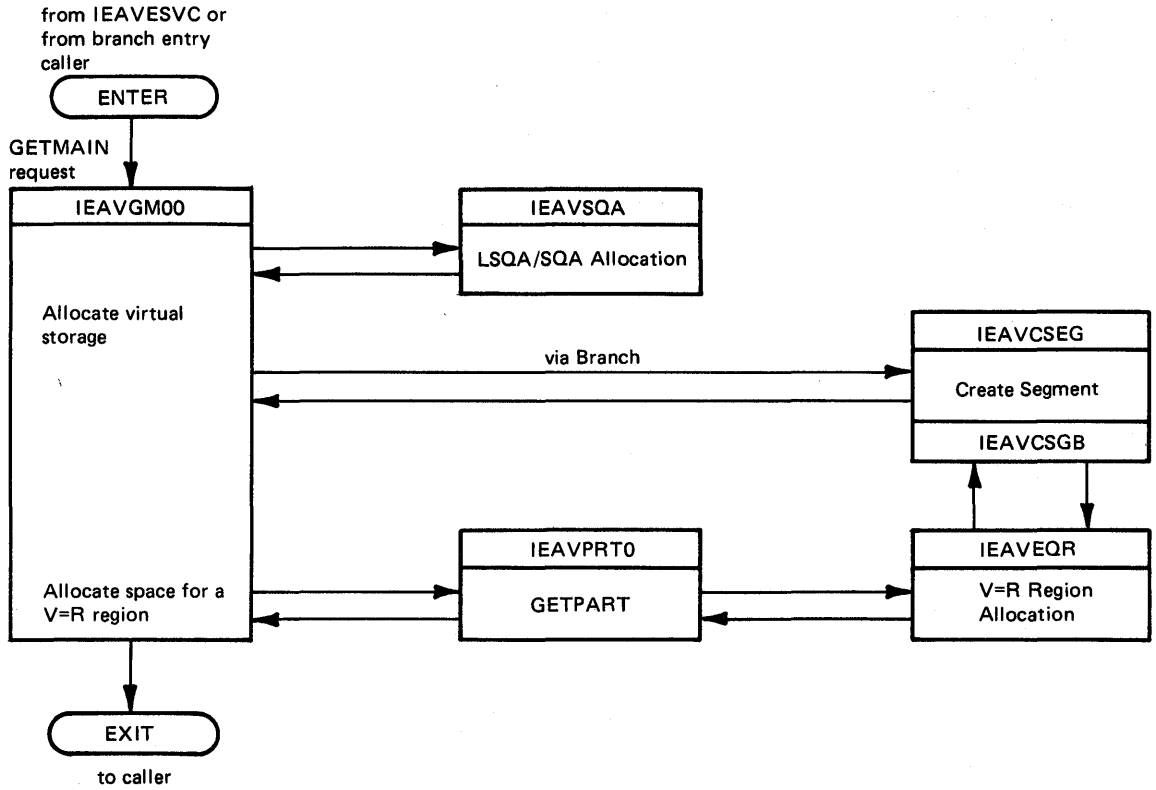


Figure 3-54. DELETE, IDENTIFY, XCTL Module Flow



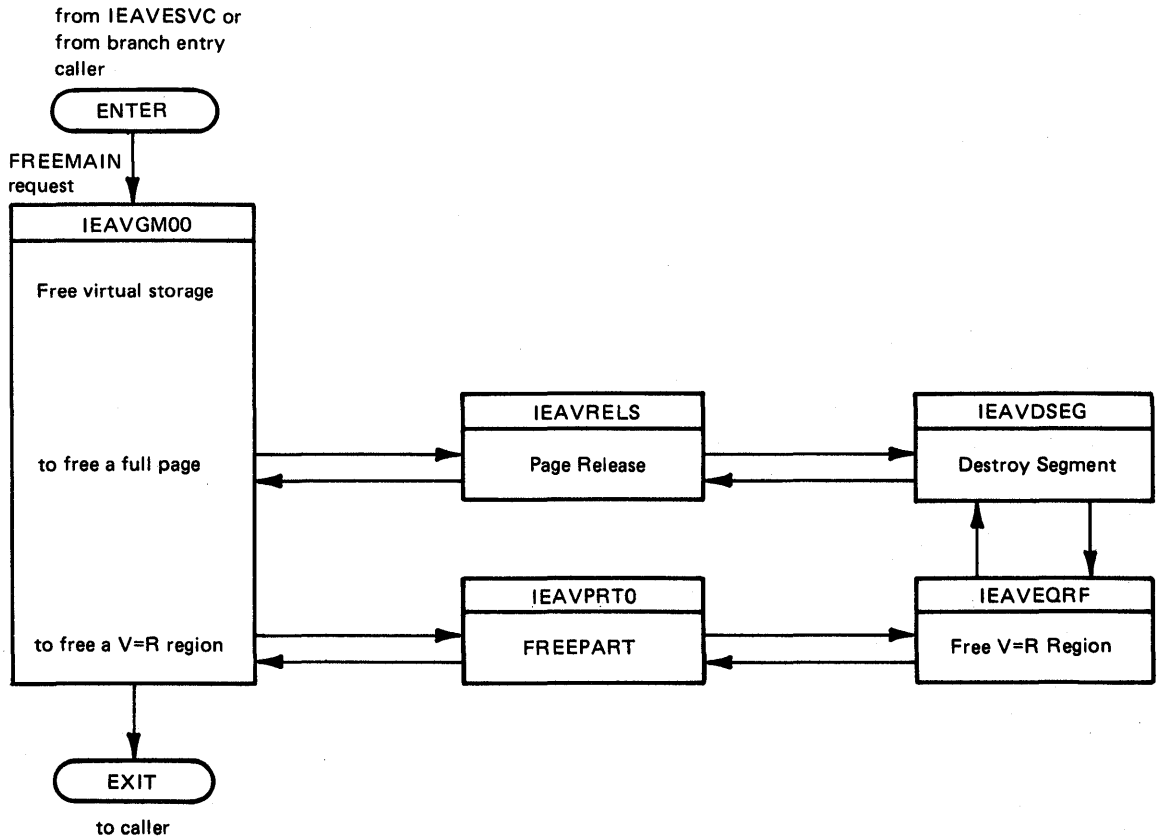


Figure 3-55. Space Allocation/Deallocation Overview Module Flow

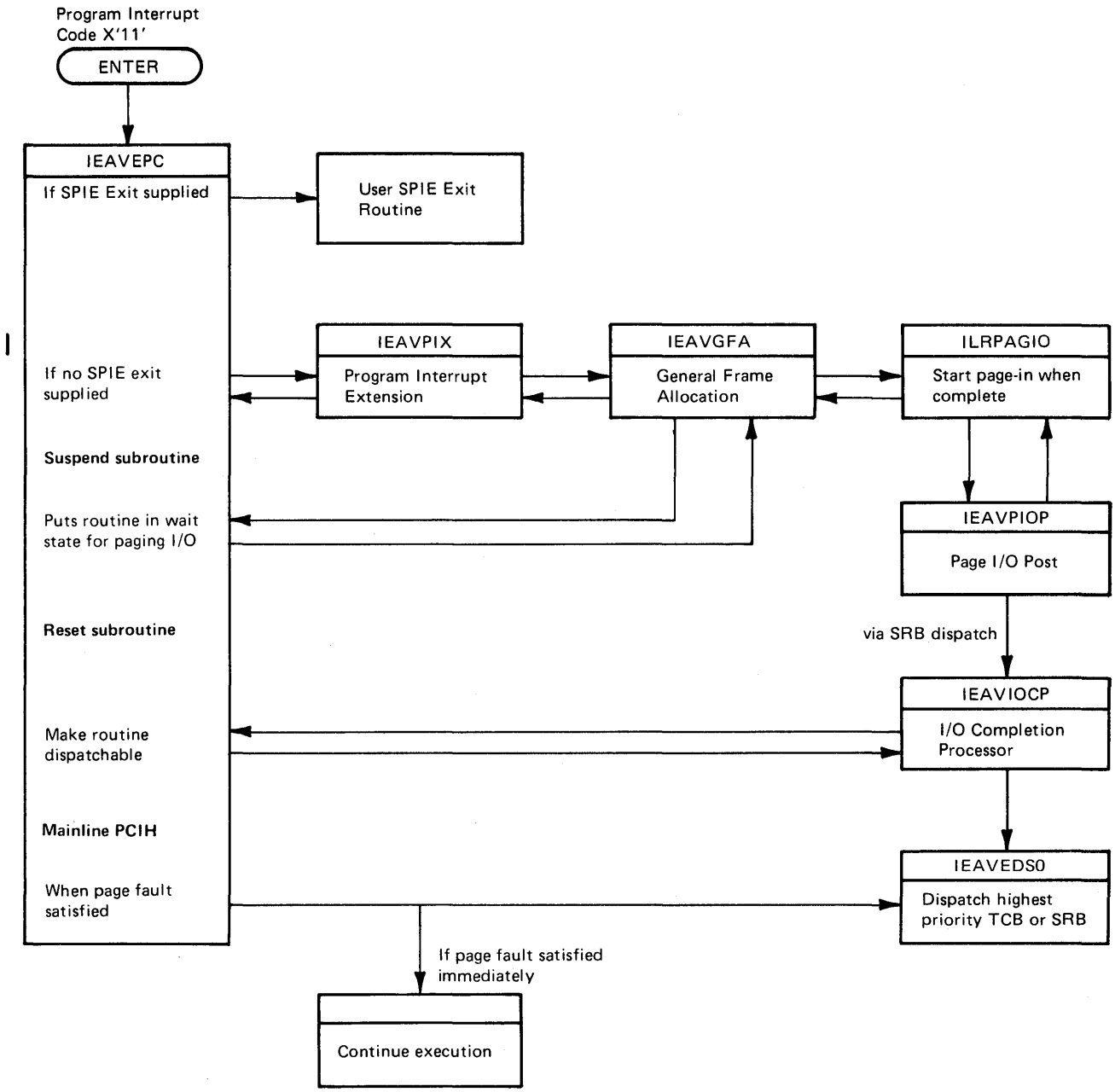


Figure 3-56. Page Fault Processing Overview Module Flow

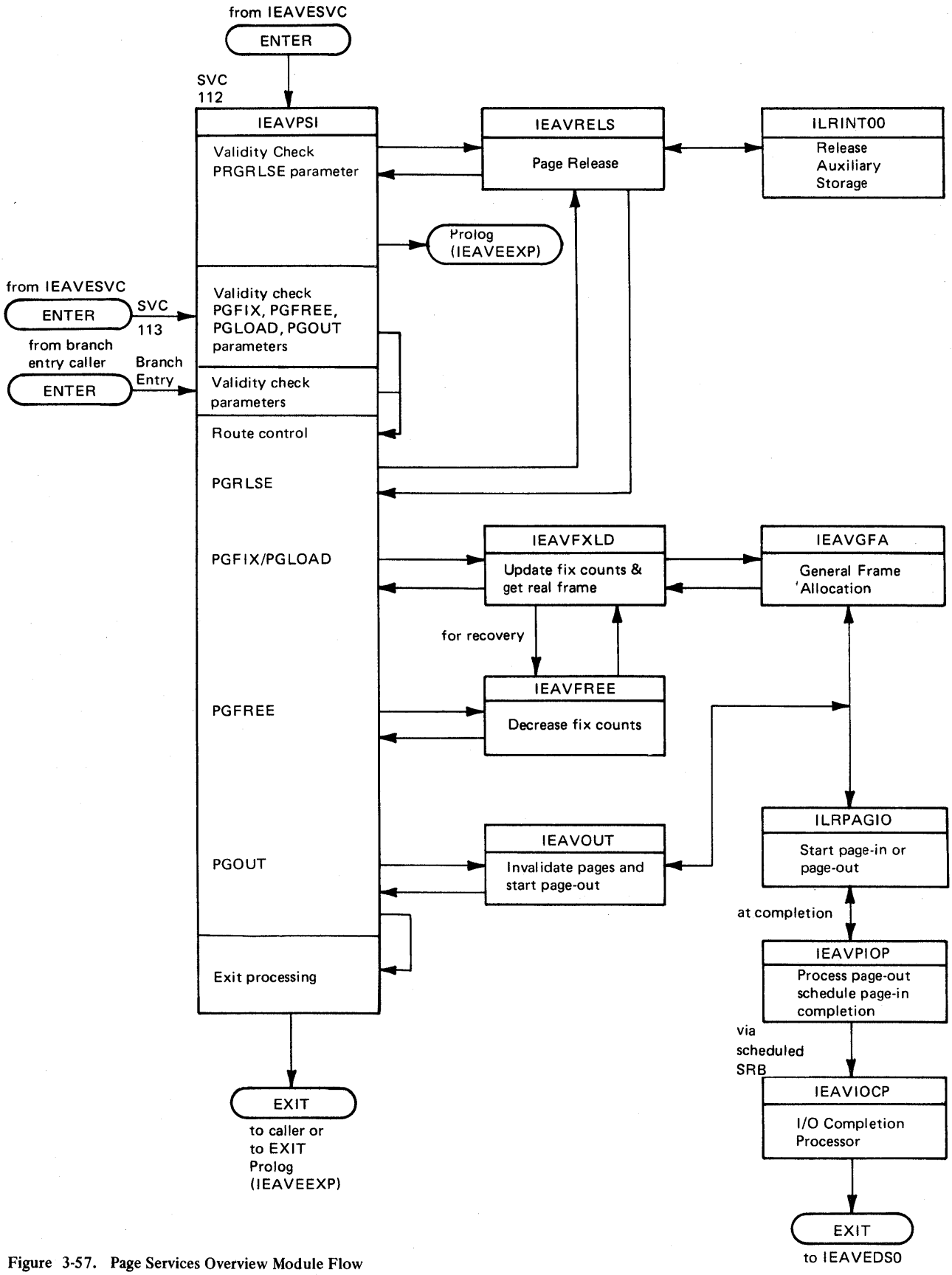


Figure 3-57. Page Services Overview Module Flow

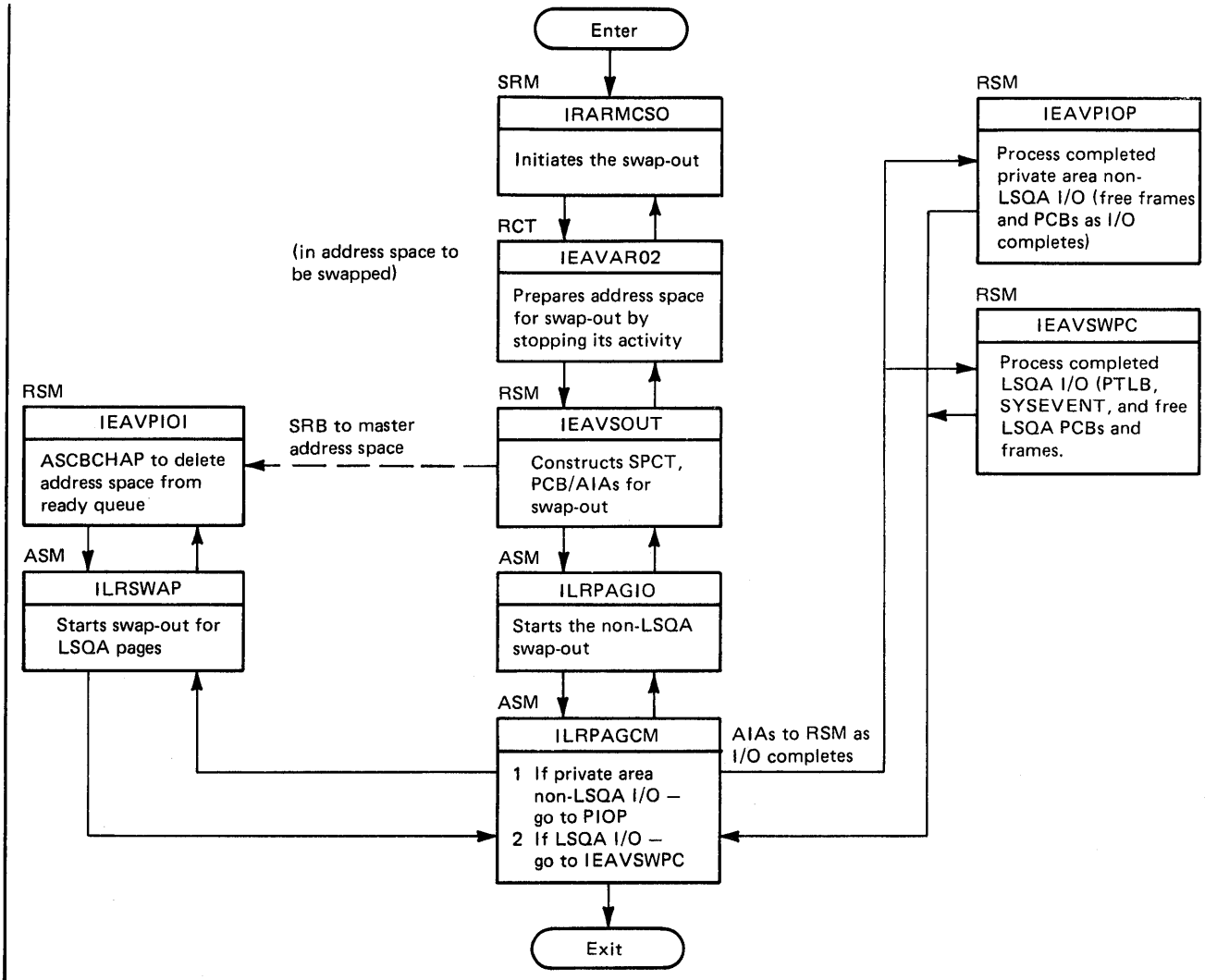


Figure 3-58A. Swap-out Overview

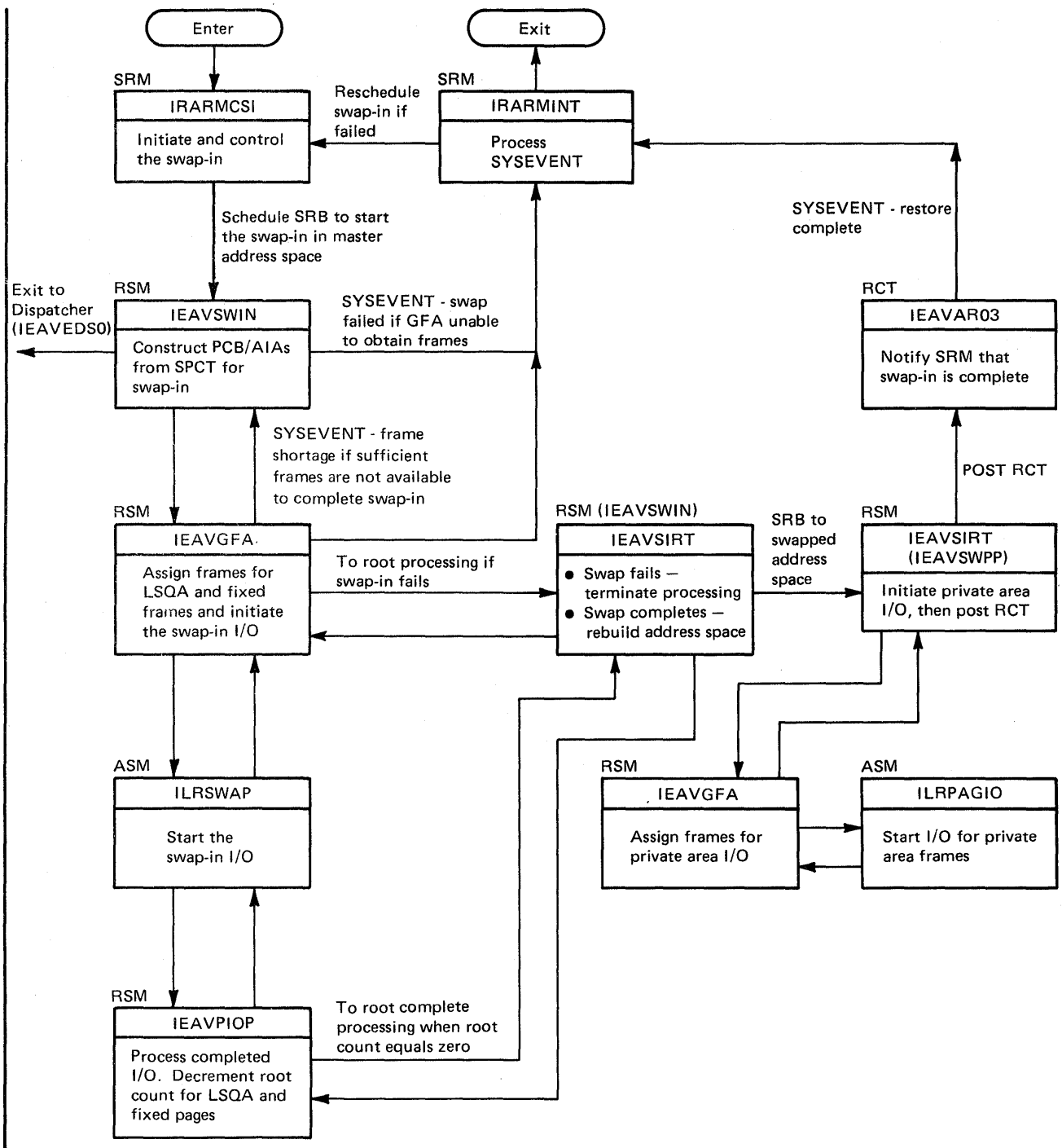


Figure 3-58B. Swap-in Overview

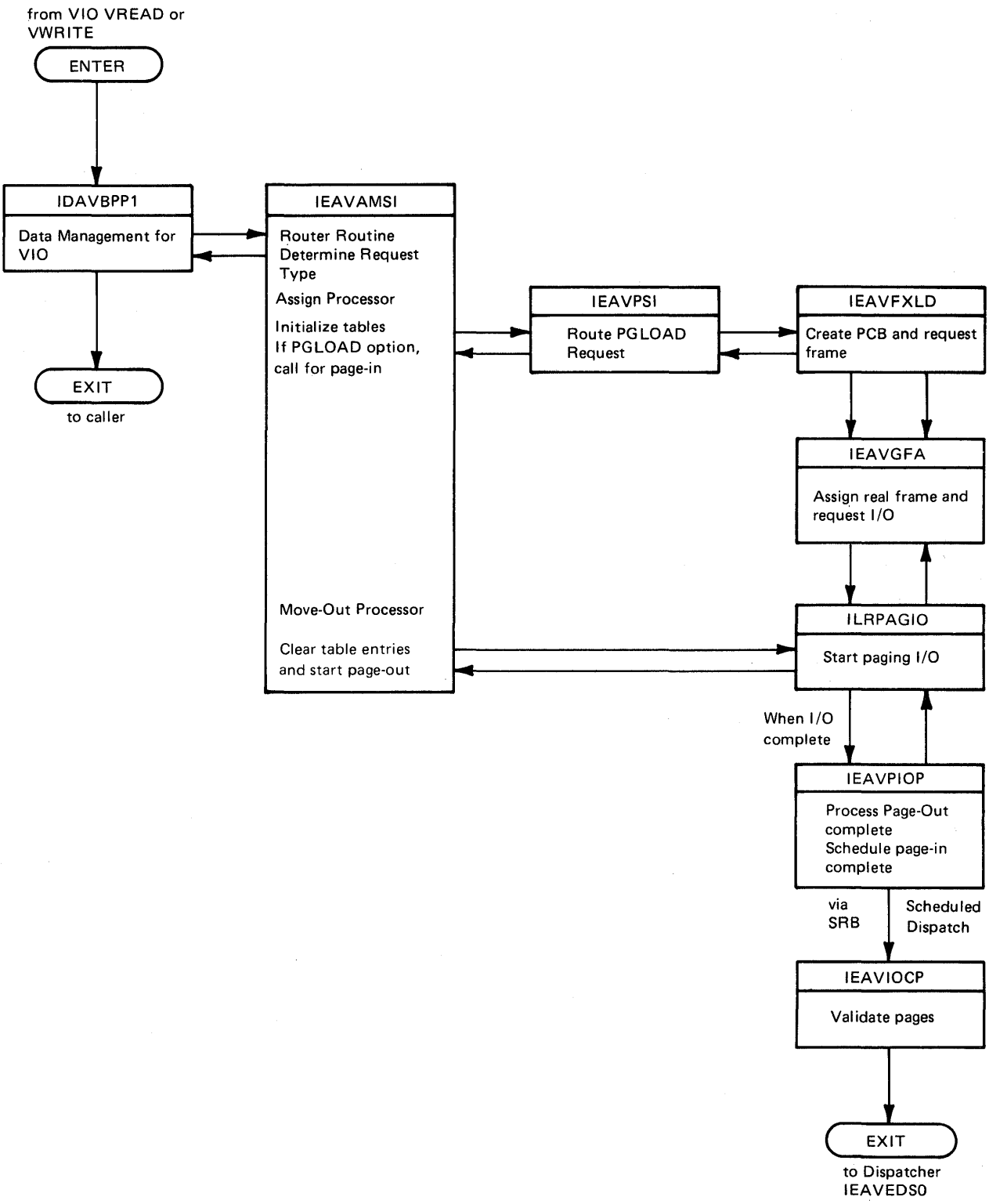


Figure 3-59. VIO Services Overview Module Flow

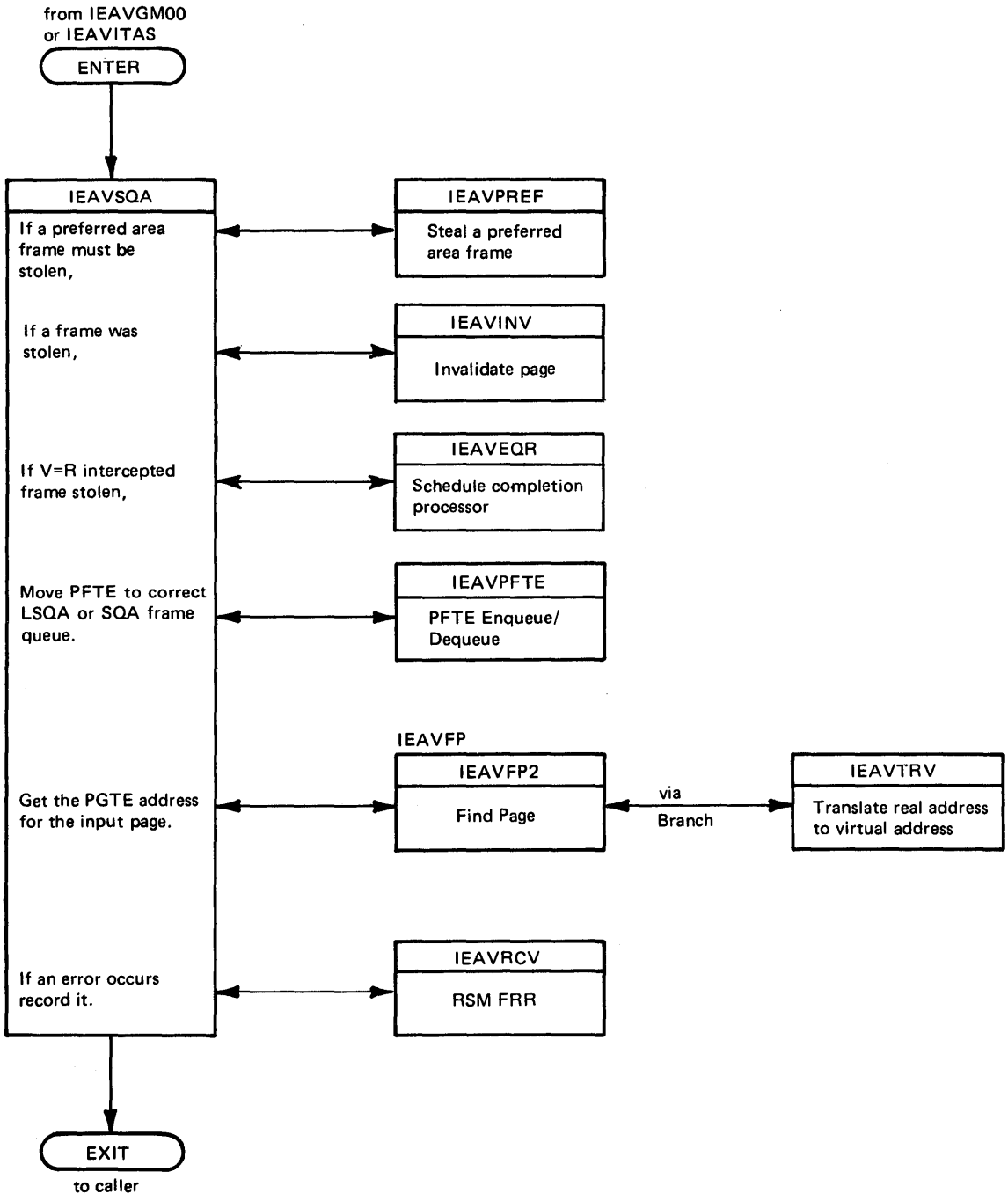


Figure 3-60. RSM Module Flow – IEAVSQA

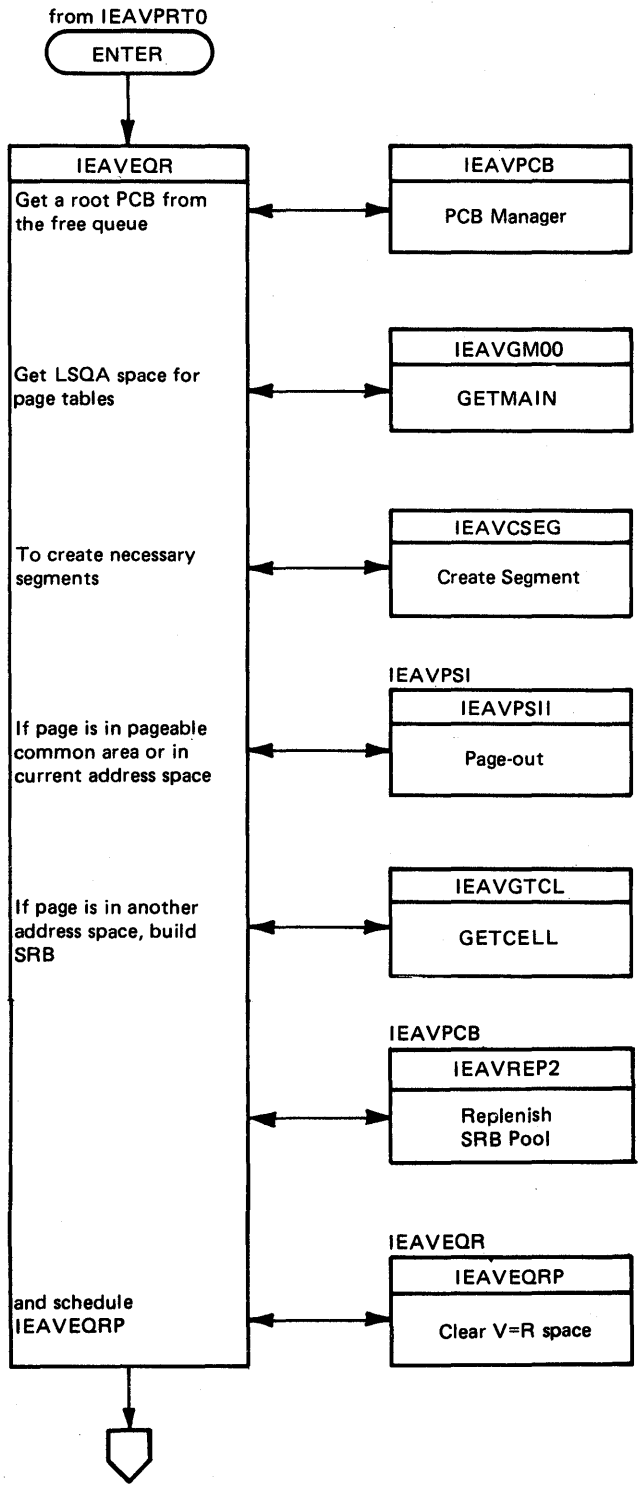


Figure 3-61. RSM Module Flow – IEAVEQR Initial Region Allocation (Part 1 of 2)

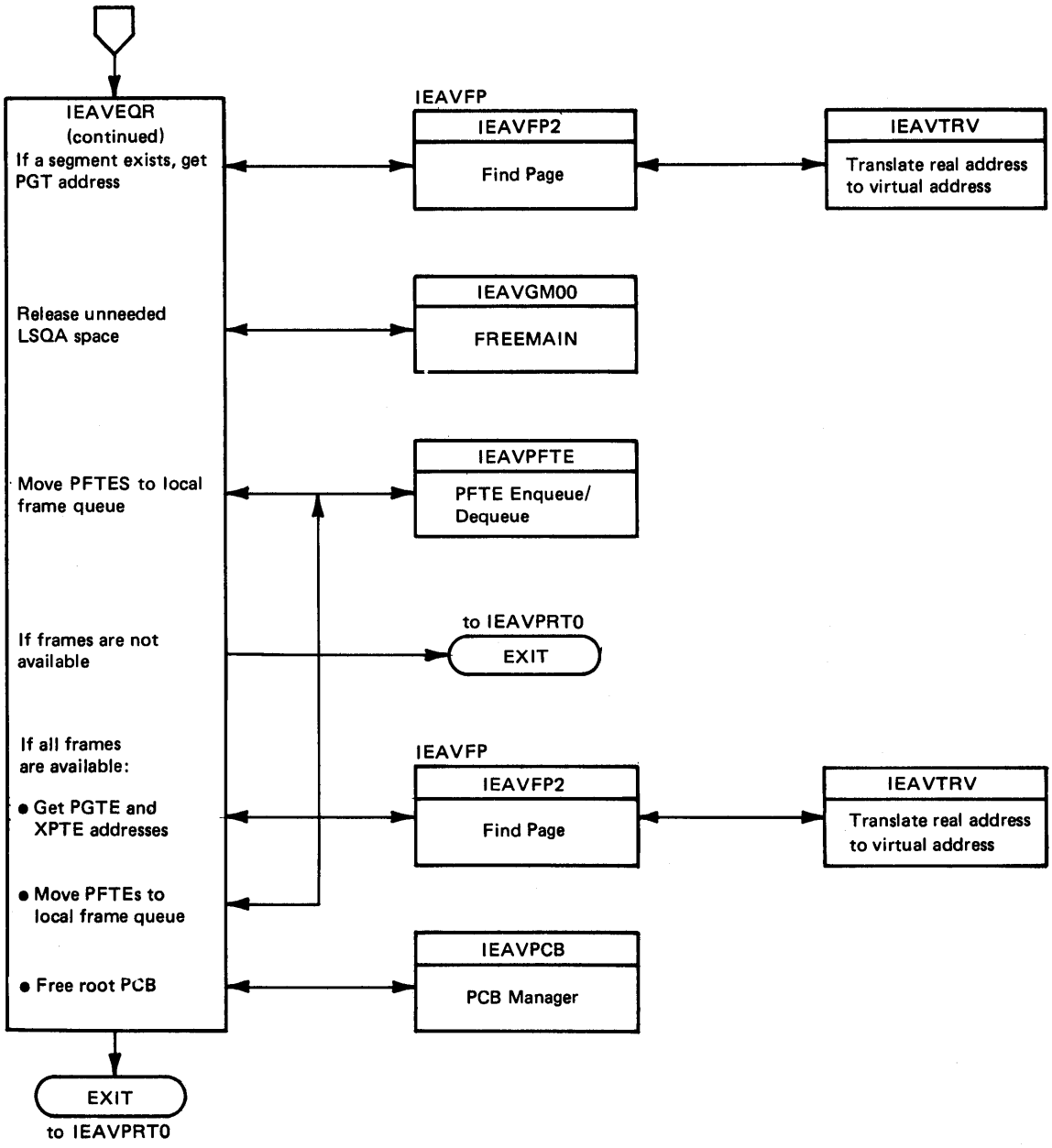


Figure 3-61. RSM Module Flow – IEAVEQR Initial Region Allocation (Part 2 of 2)

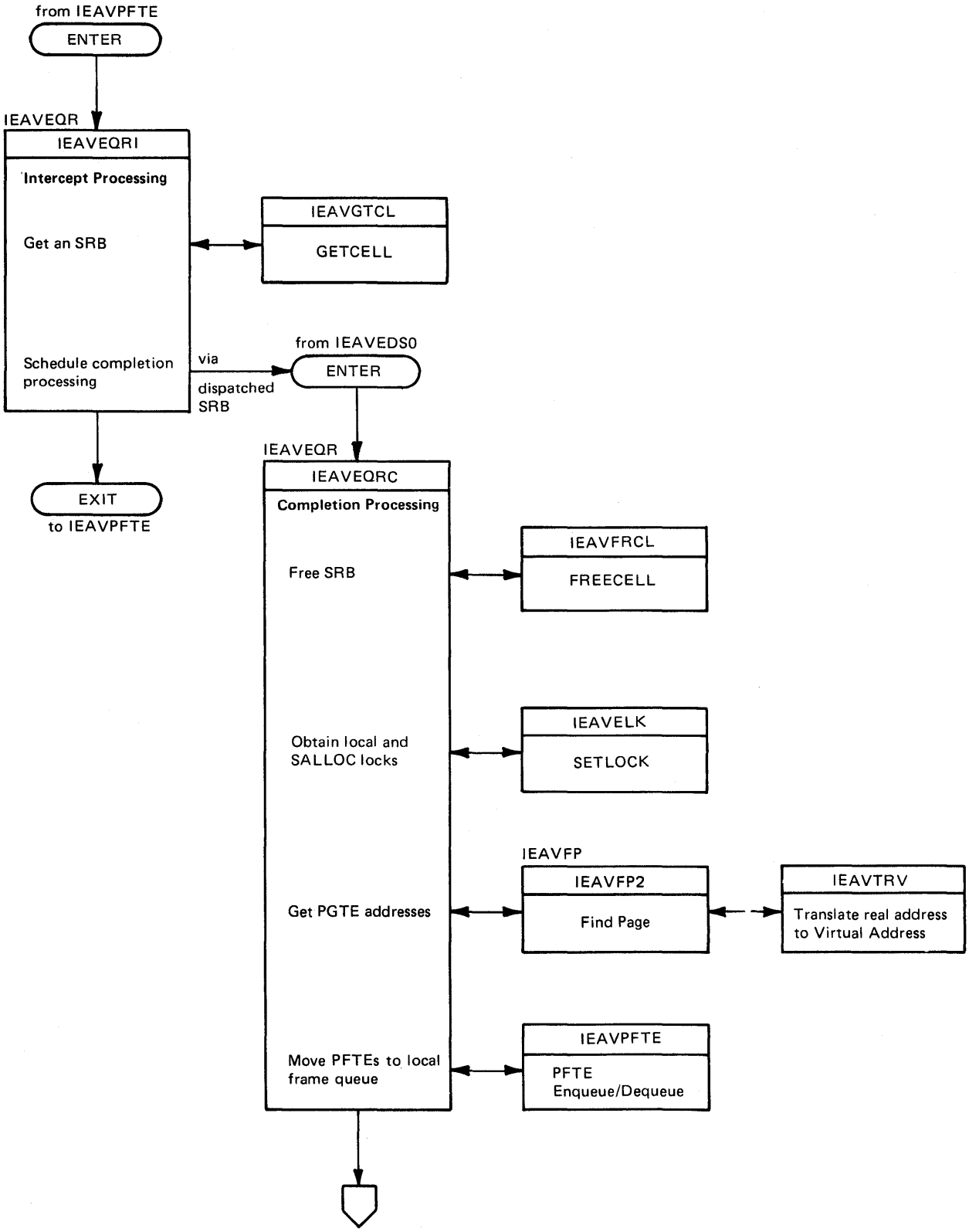


Figure 3-62. RSM Module Flow – IEAVEQR Asynchronous Completion and V=R Clearing (Part 1 of 2)

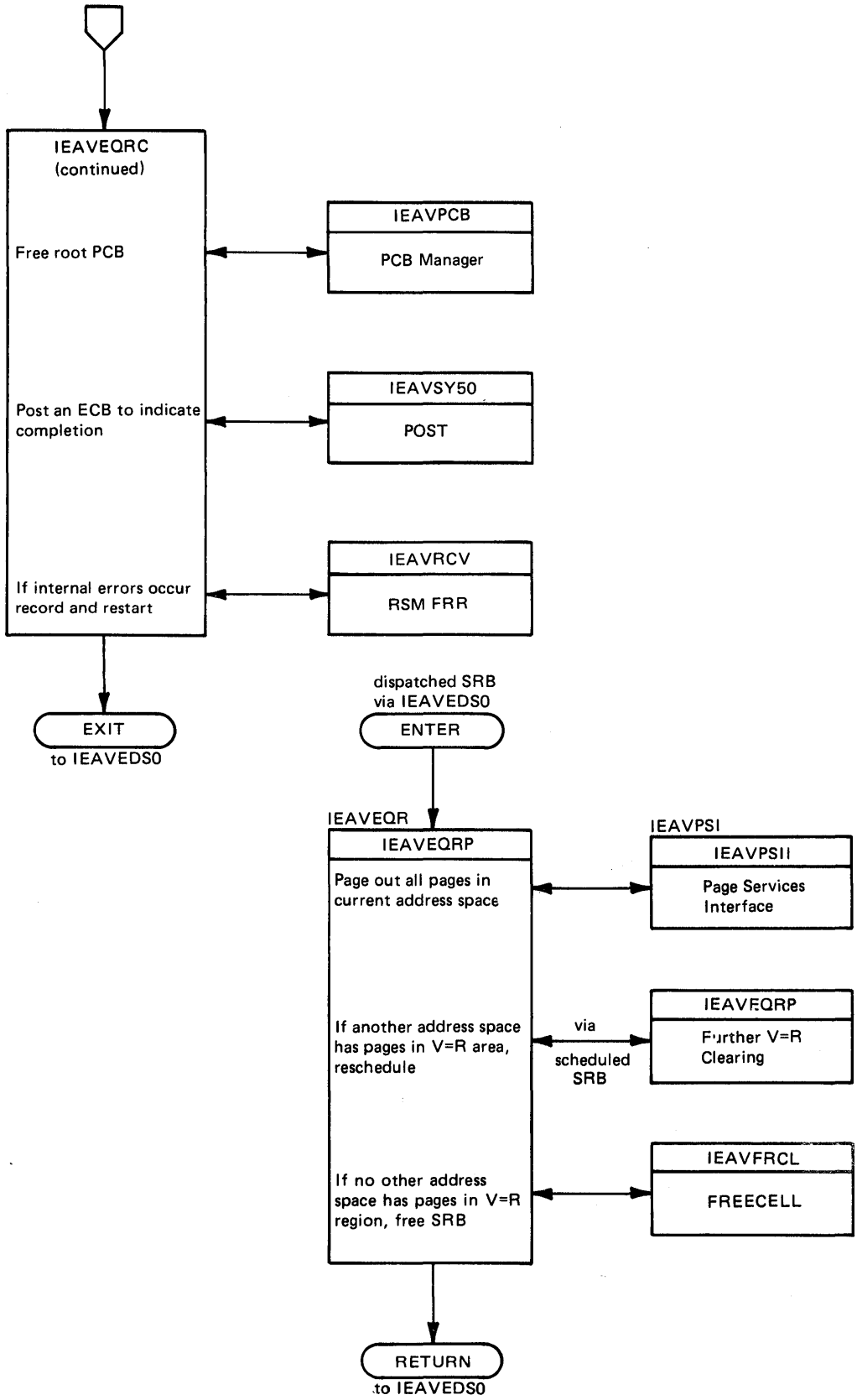


Figure 3-62. RSM Module Flow – IEAVEQR Asynchronous Completion and V=R Clearing (Part 2 of 2)

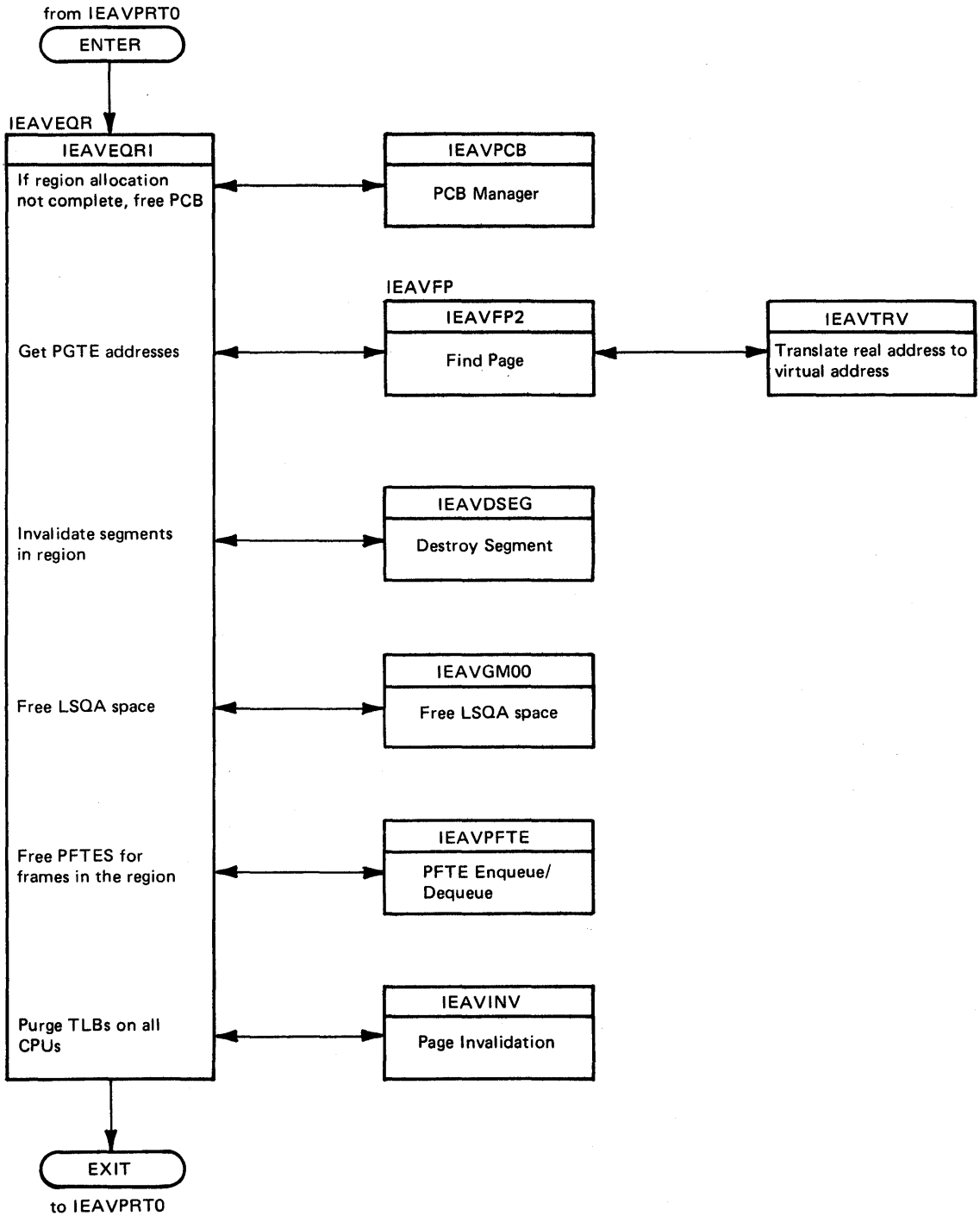


Figure 3-63. RSM Module Flow – IEAVEQR Free V=R Region

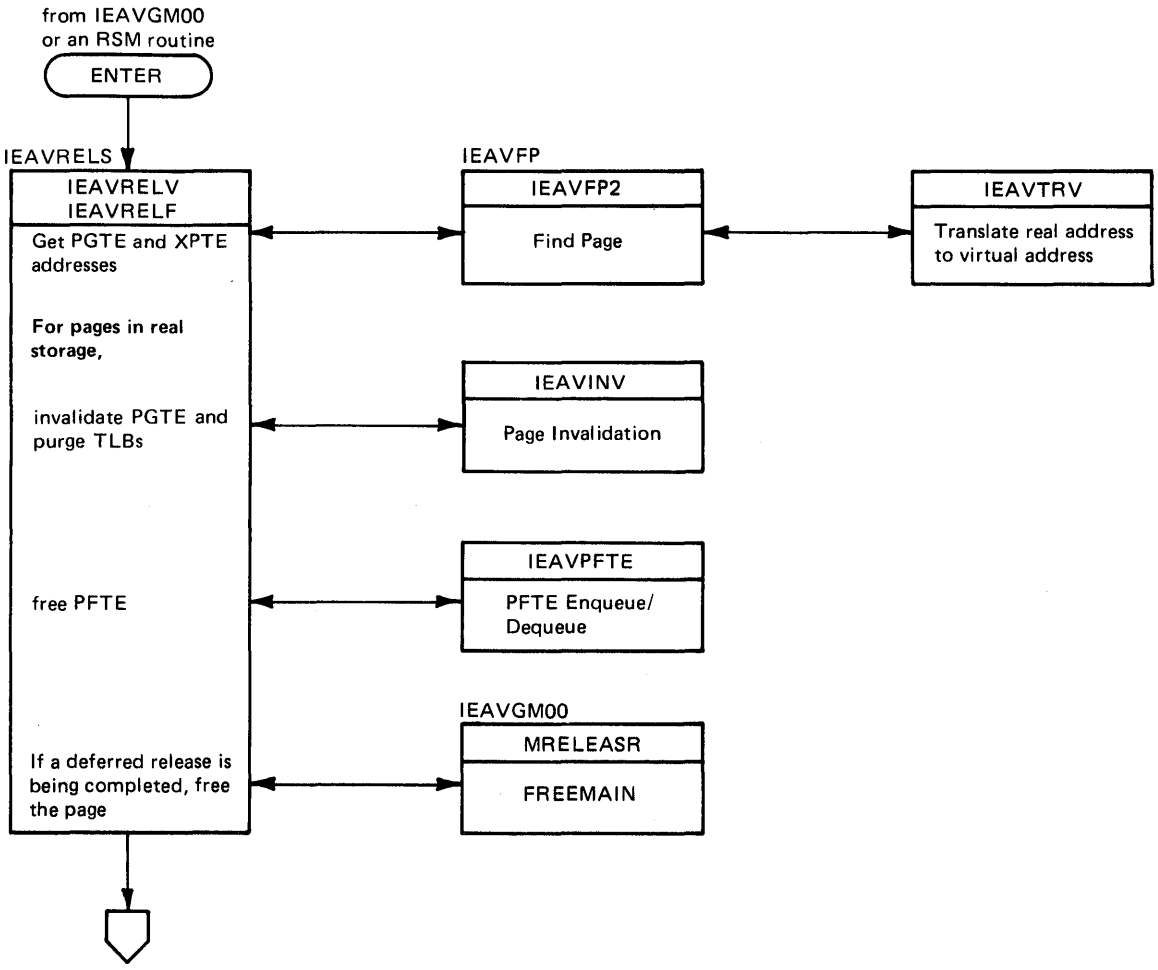


Figure 3-64. RSM Module Flow – IEAVRELS FREEMAIN Release (Part 1 of 2)

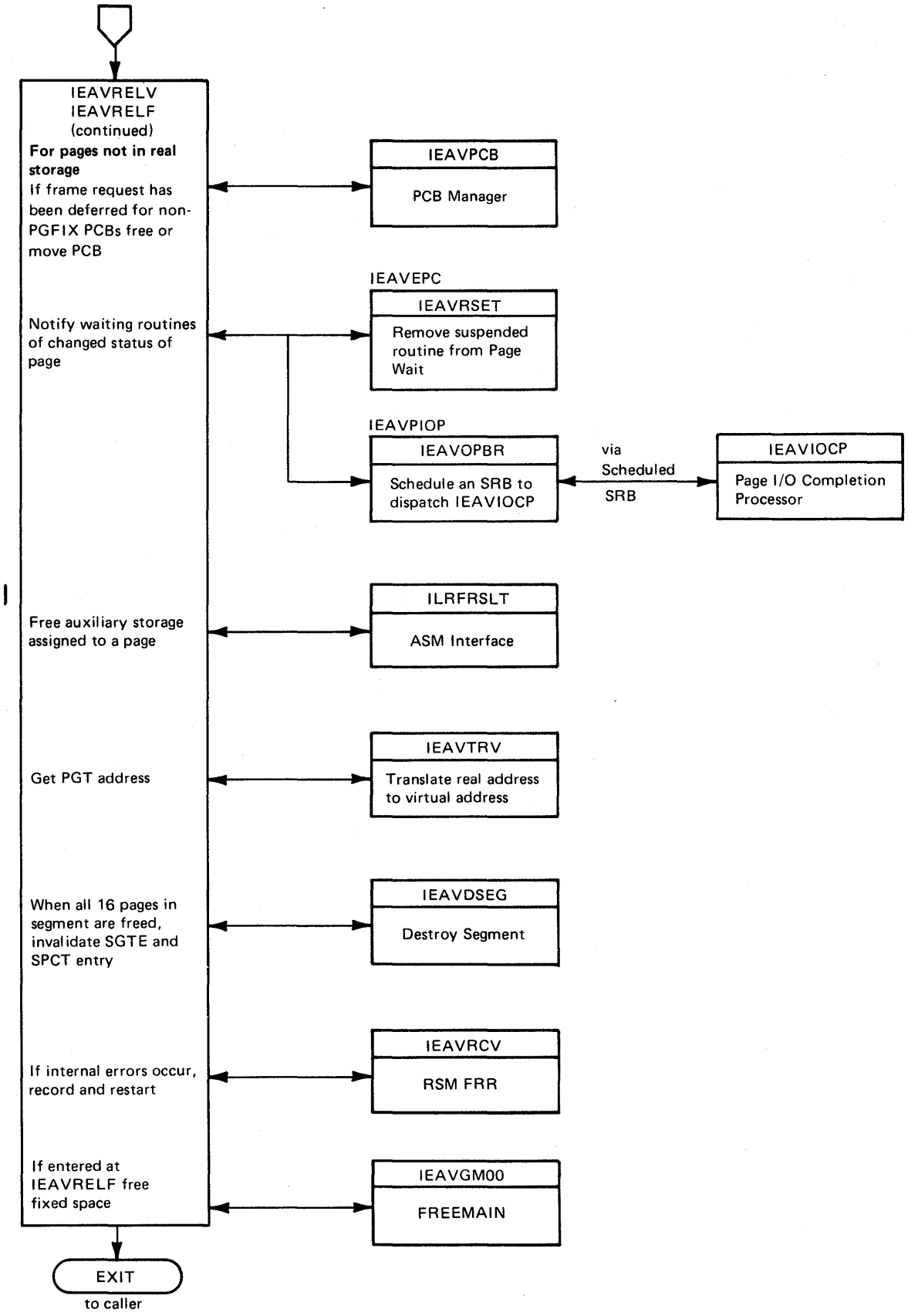


Figure 3-64. RSM Module Flow – IEAVRELS FREEMAIN Release (Part 2 of 2)

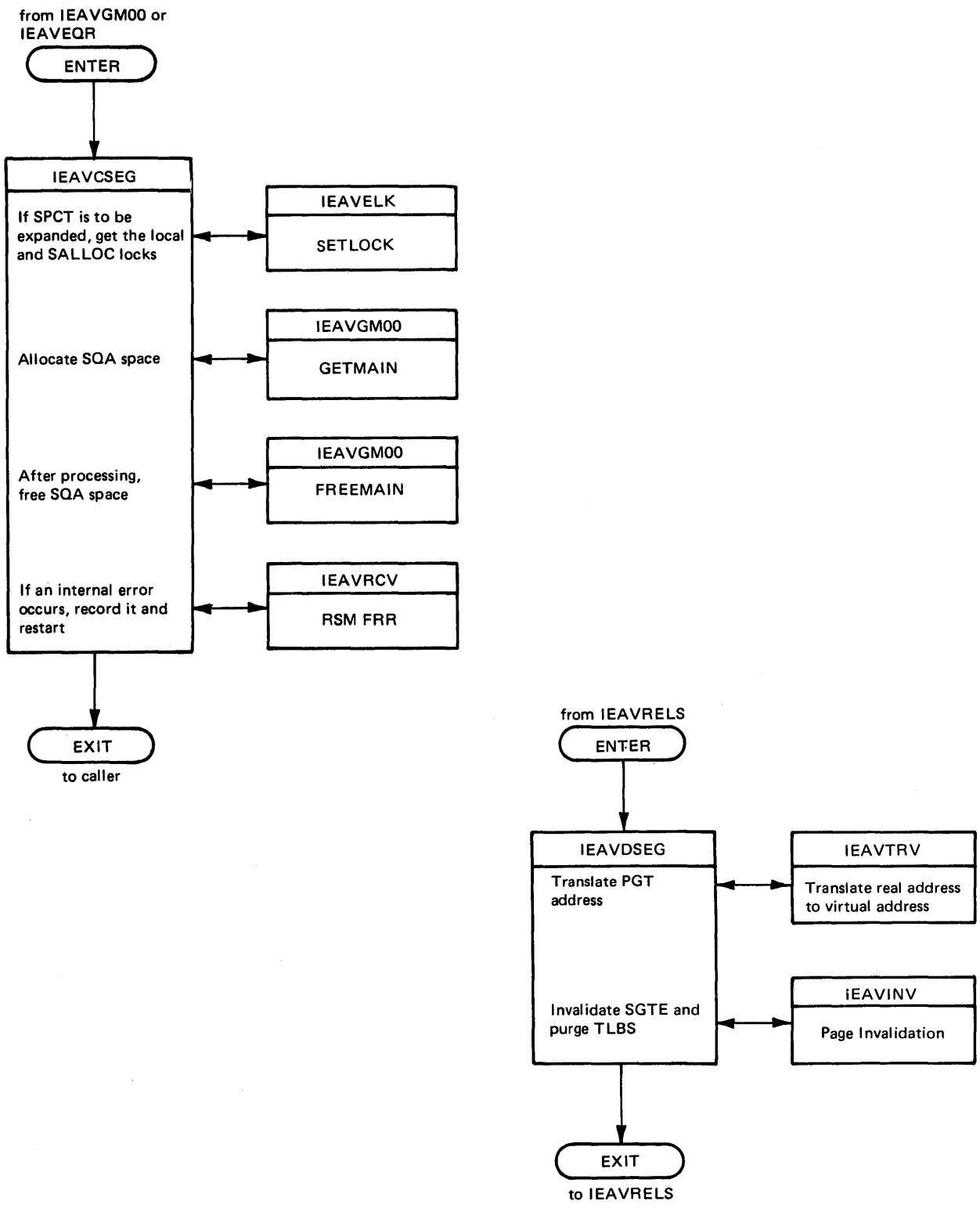


Figure 3-65. RSM Module Flow – IEAVCSEG, IEAVDSEG

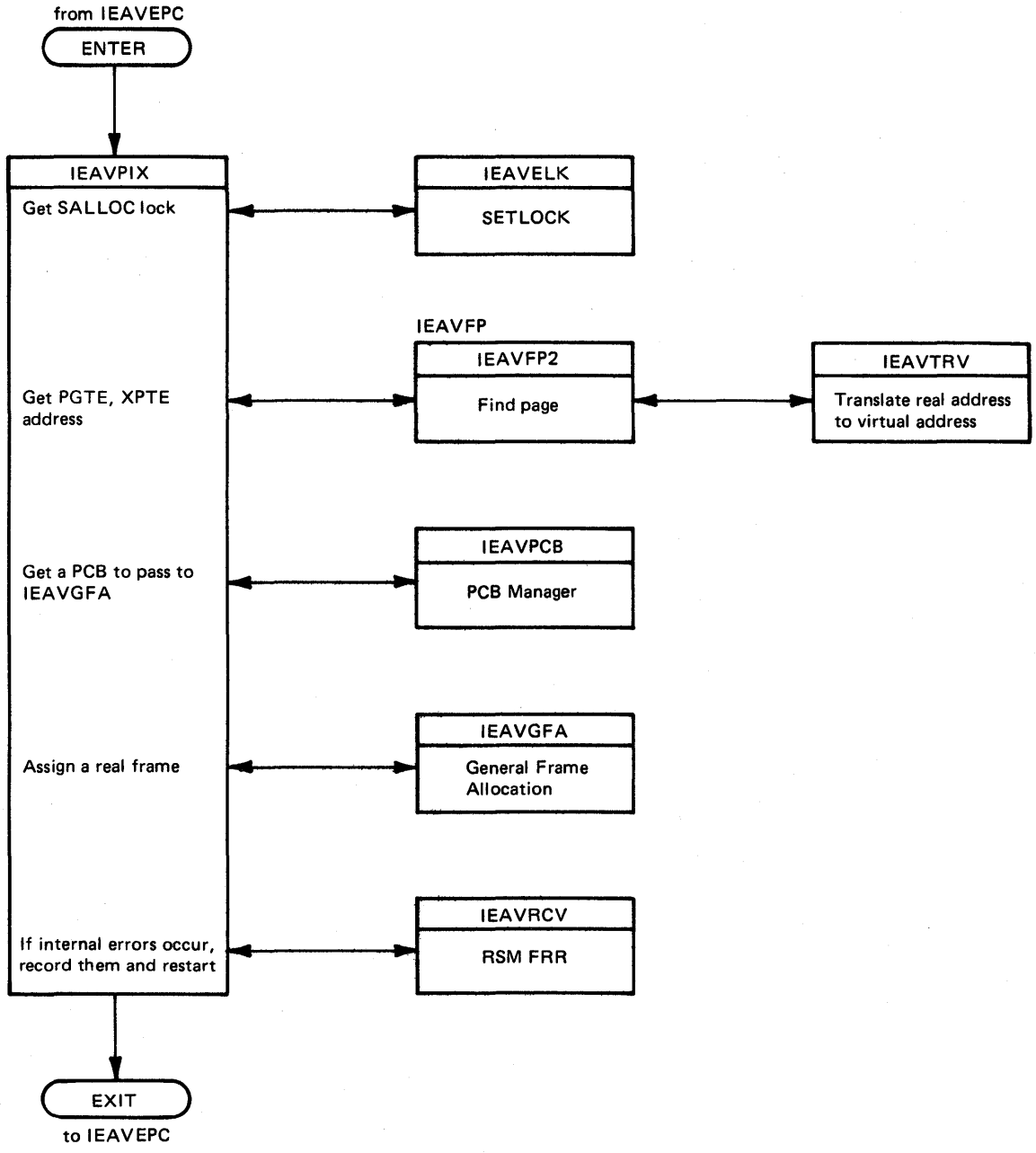


Figure 3-66. RSM Module Flow – IEAVPIX

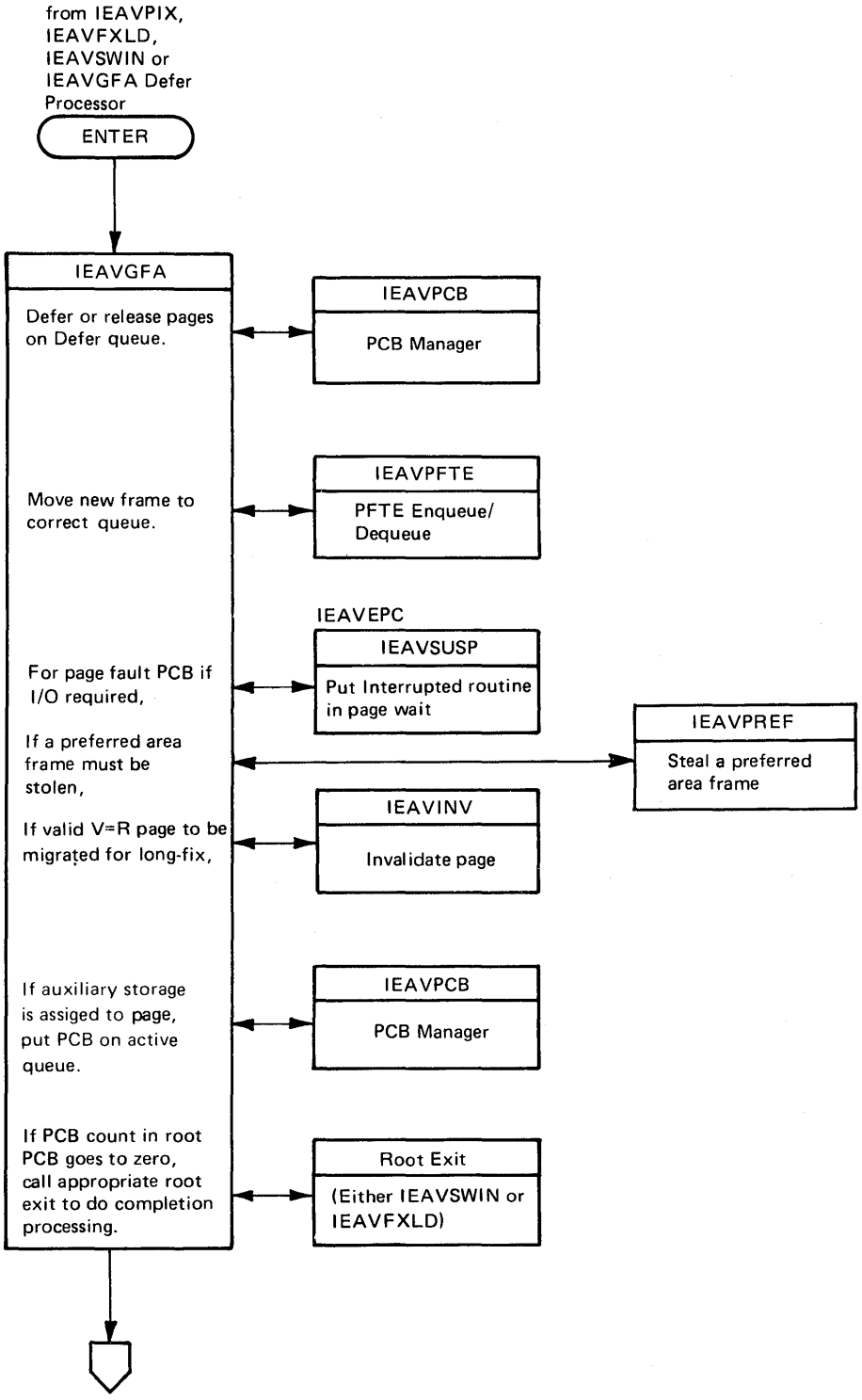


Figure 3-67. RSM Module Flow – IEAVGFA (Part 1 of 2)

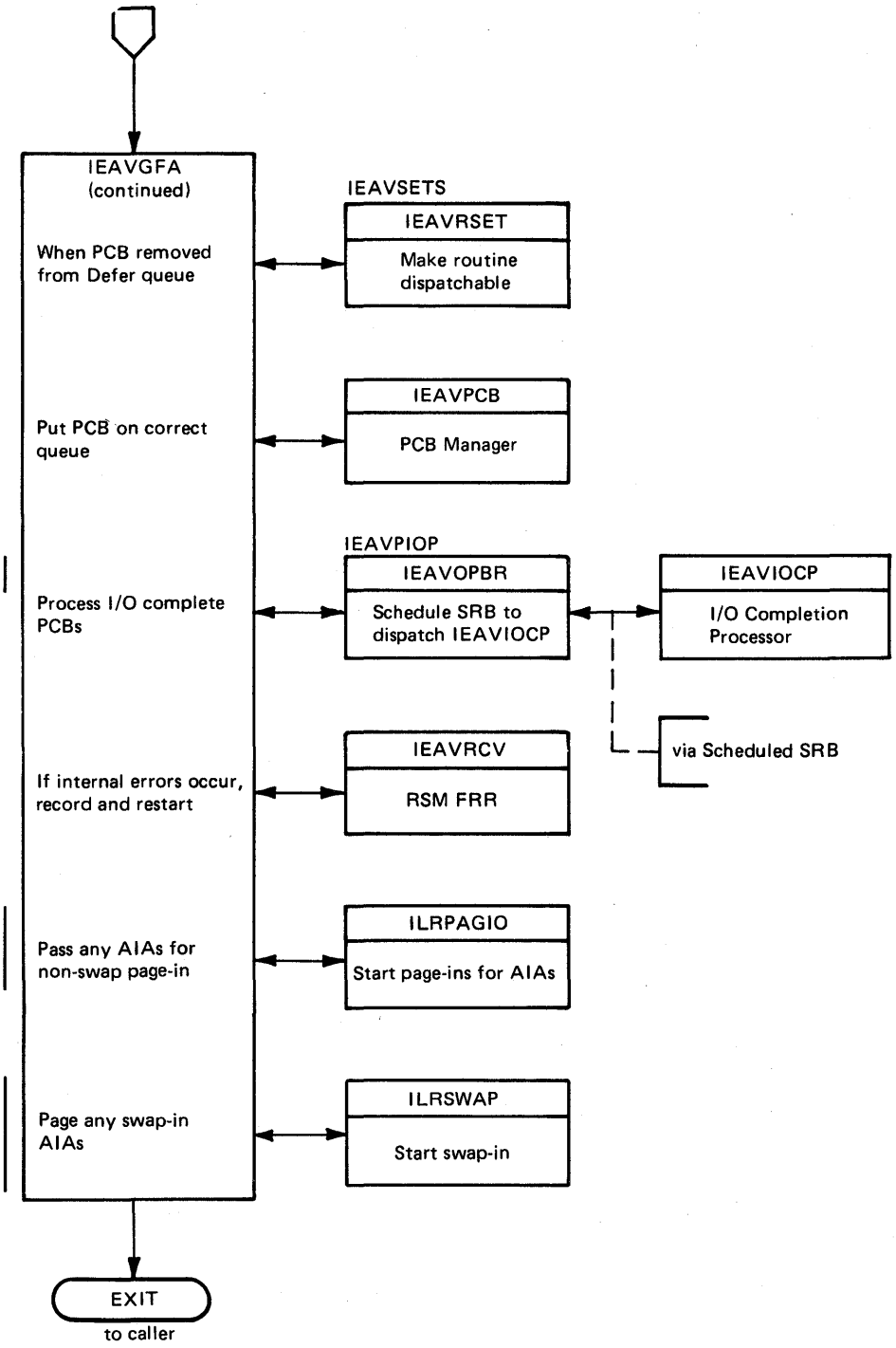


Figure 3-67. RSM Module Flow – IEAVGFA (Part 2 of 2)

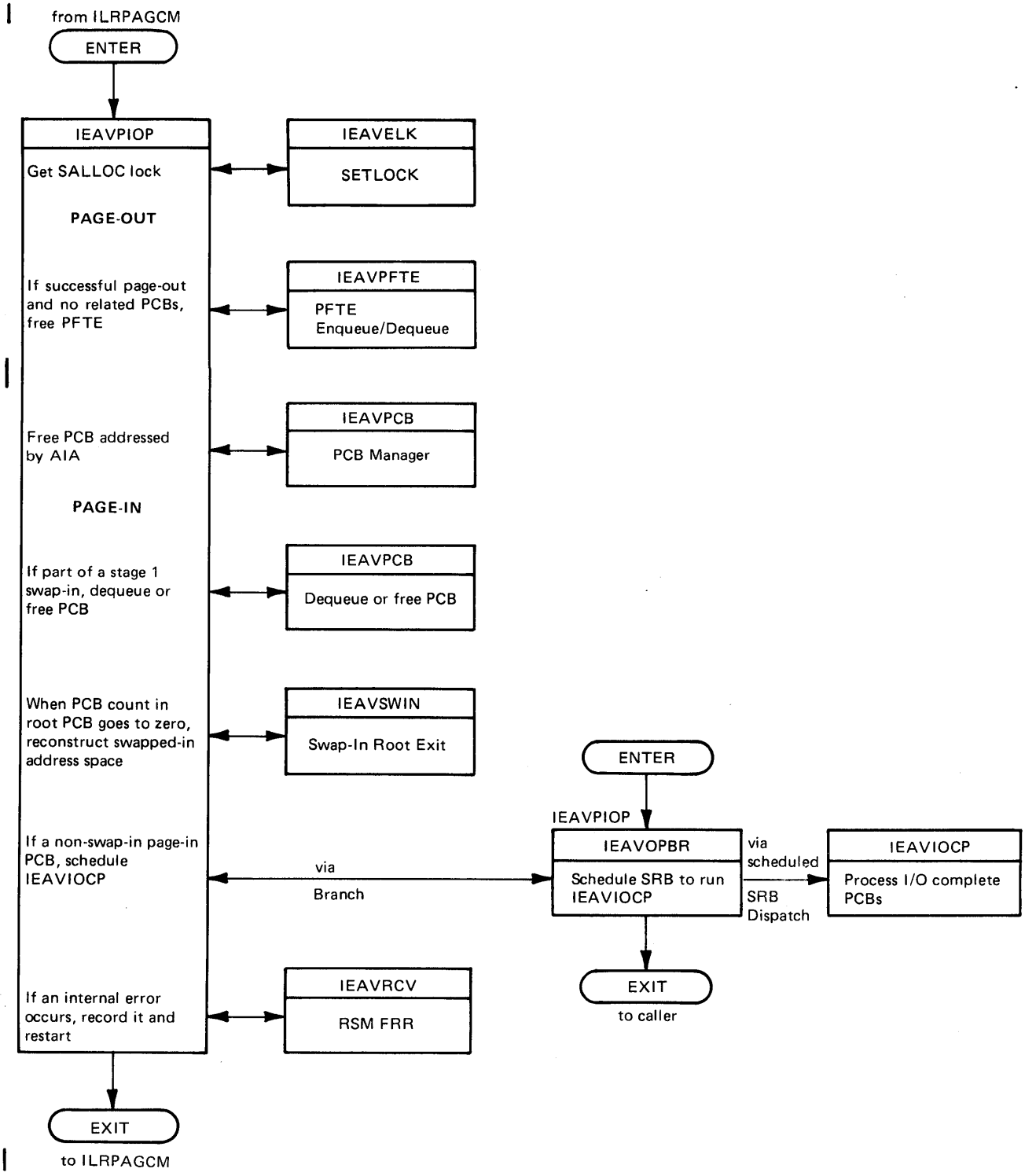


Figure 3-68. RSM Module Flow – IEAVPIOP

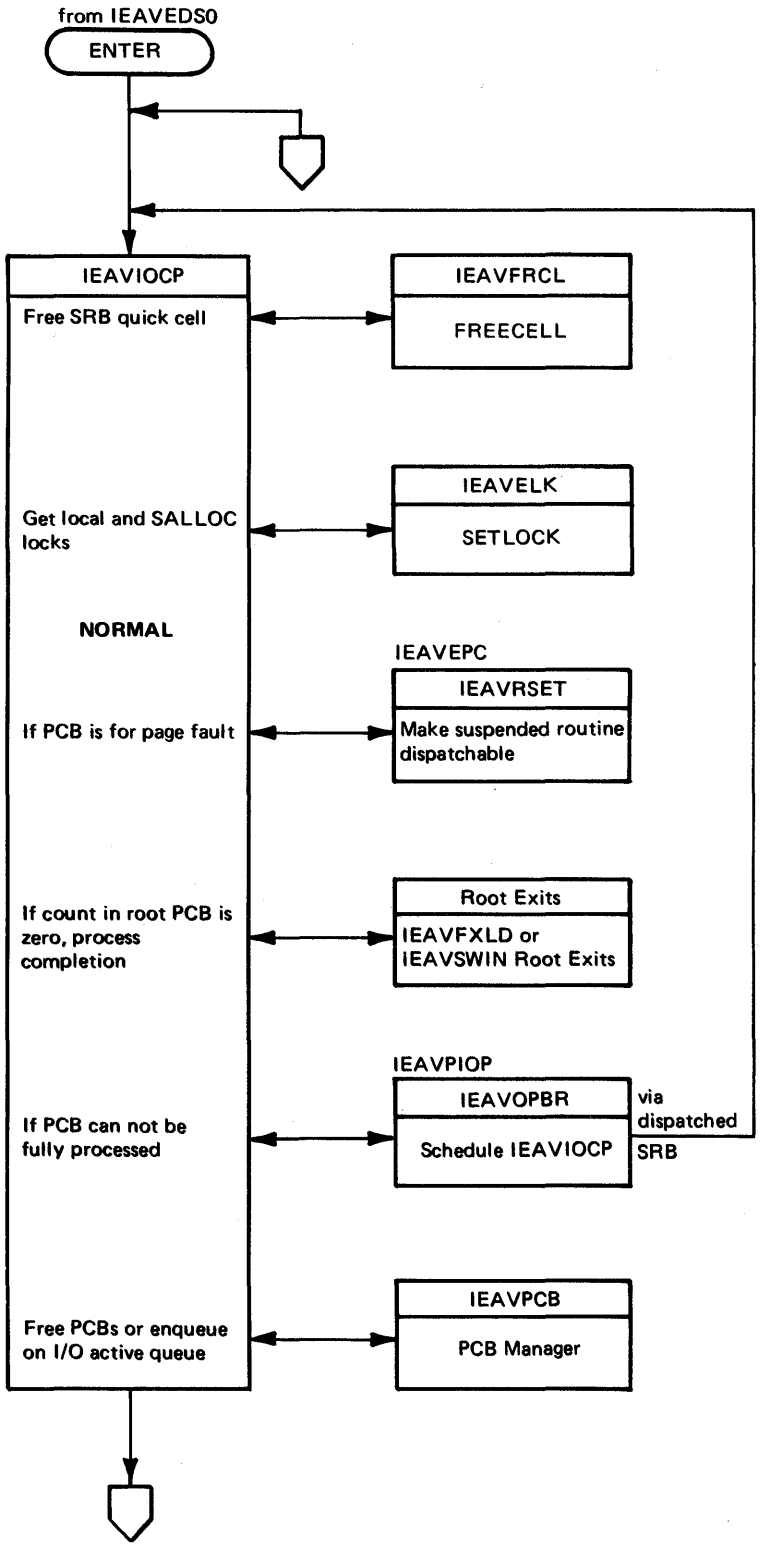


Figure 3-69. RSM Module Flow – IEAVIOCP (Part 1 of 2)

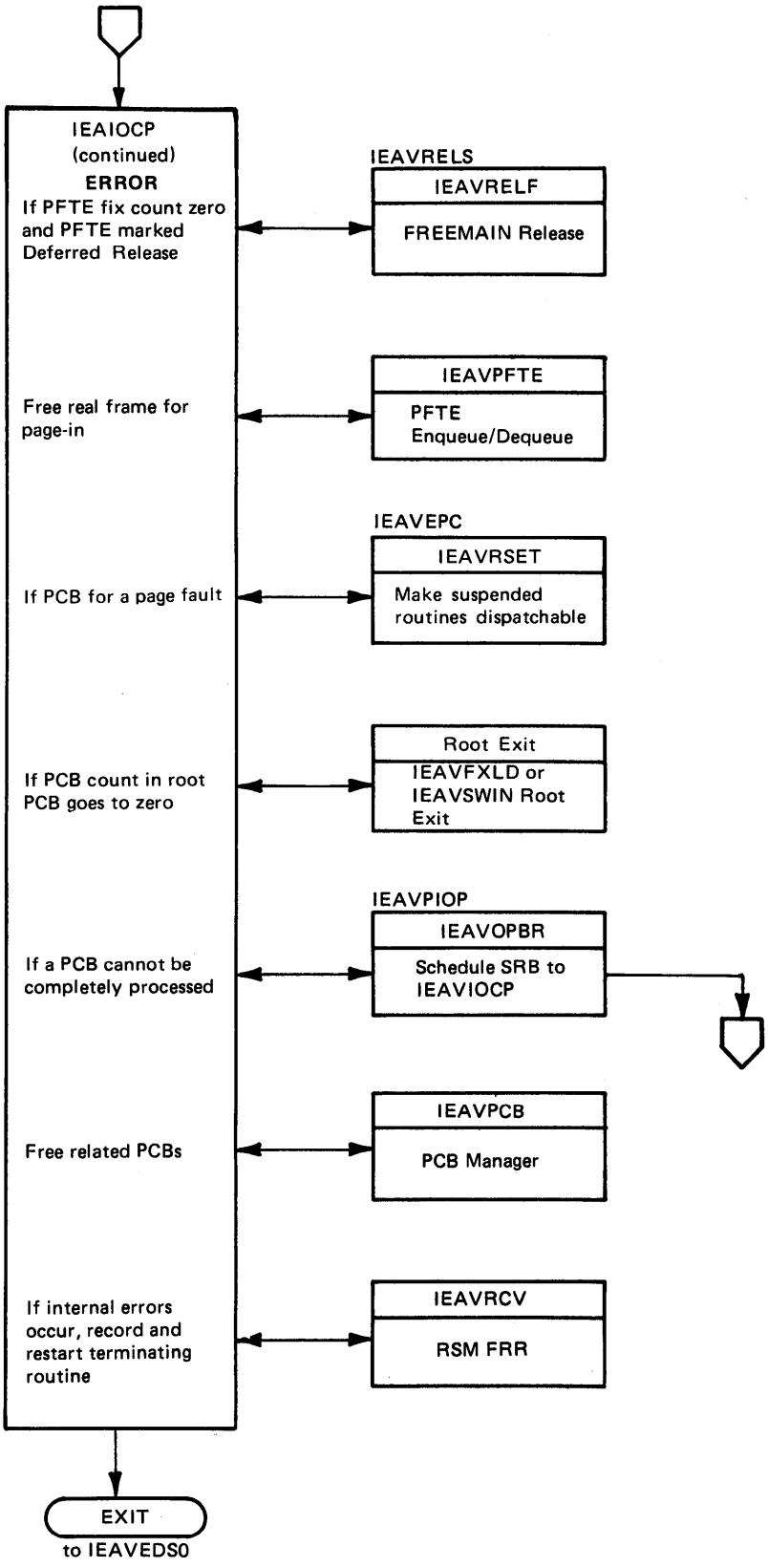


Figure 3-69. RSM Module Flow – IEAVIOCP (Part 2 of 2)

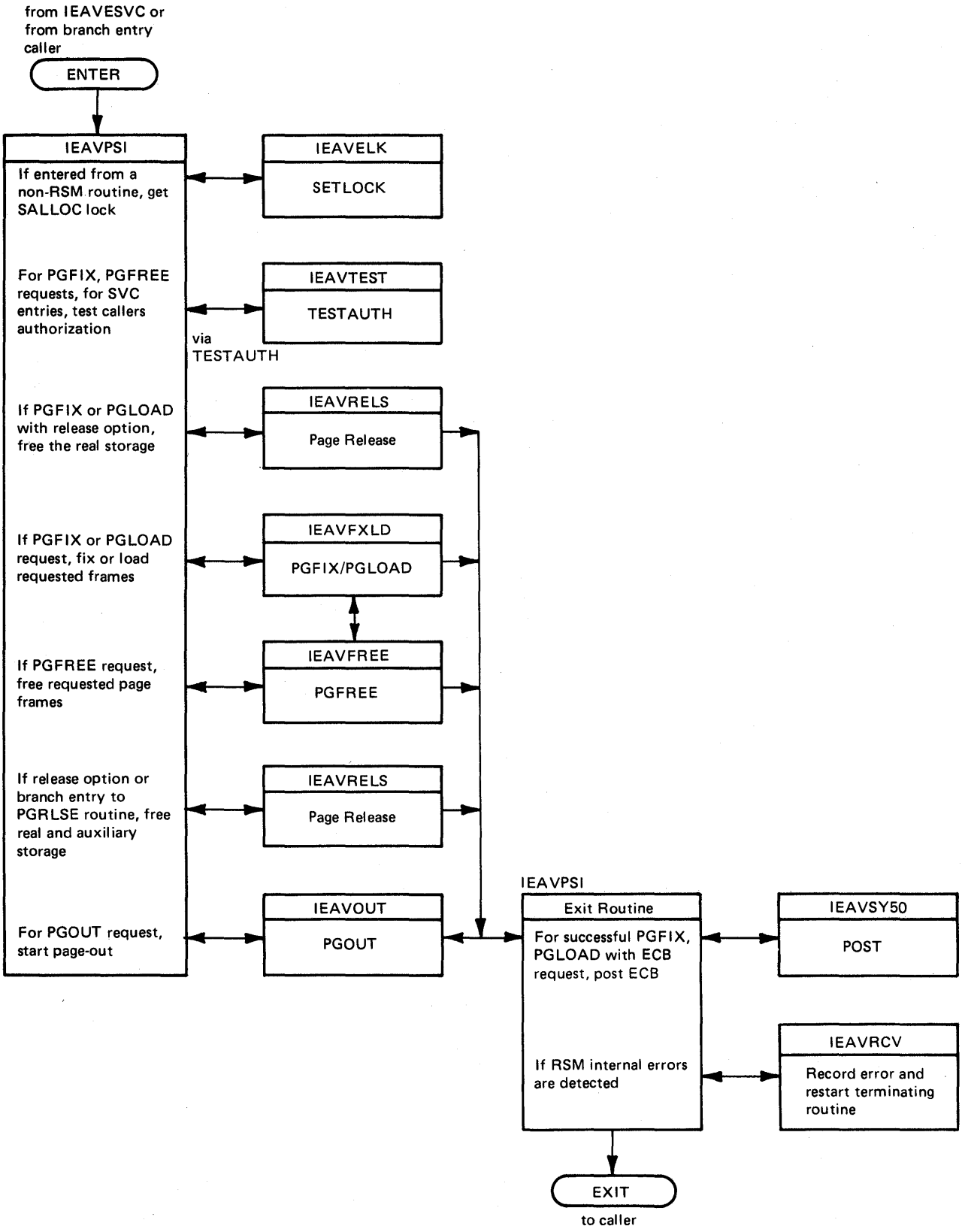


Figure 3-70. RSM Module Flow – IEAVPSI

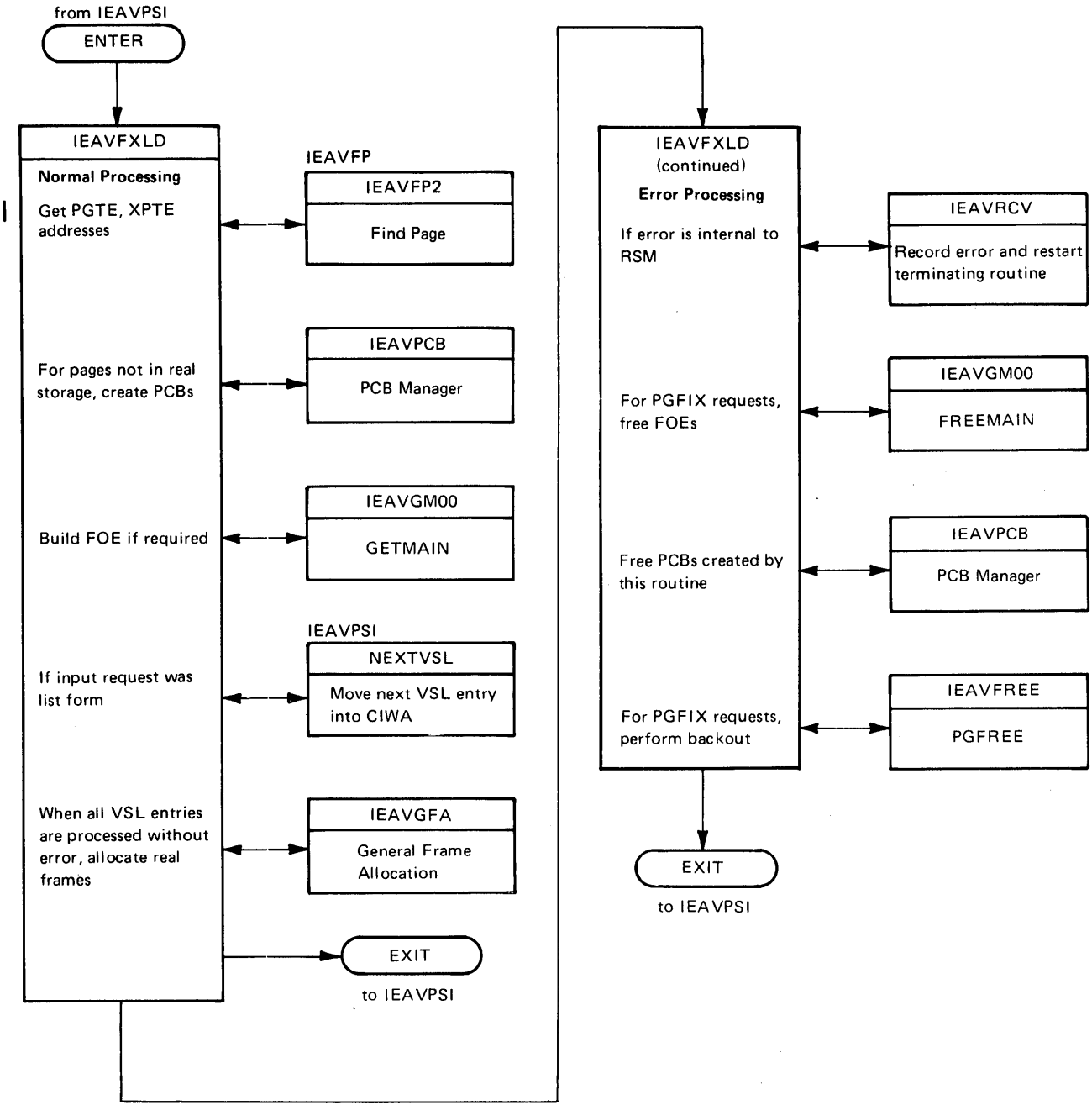


Figure 3-71. RSM Module Flow – IEAVFXLD and IEAVFXLD Root Exit (Part 1 of 2)

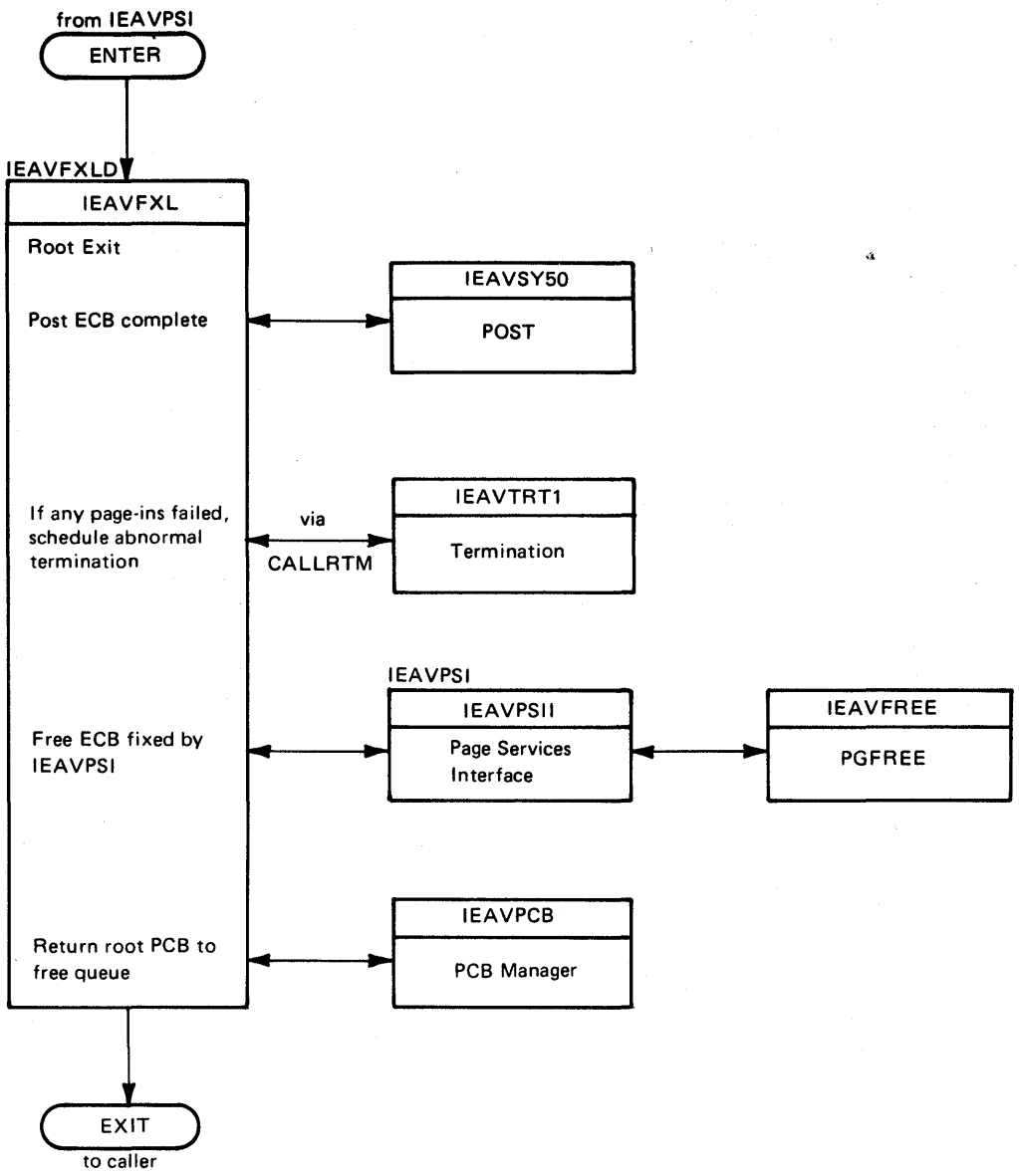


Figure 3-71. RSM Module Flow – IEAVFXLD and IEAVFXLD Root Exit (Part 2 of 2)

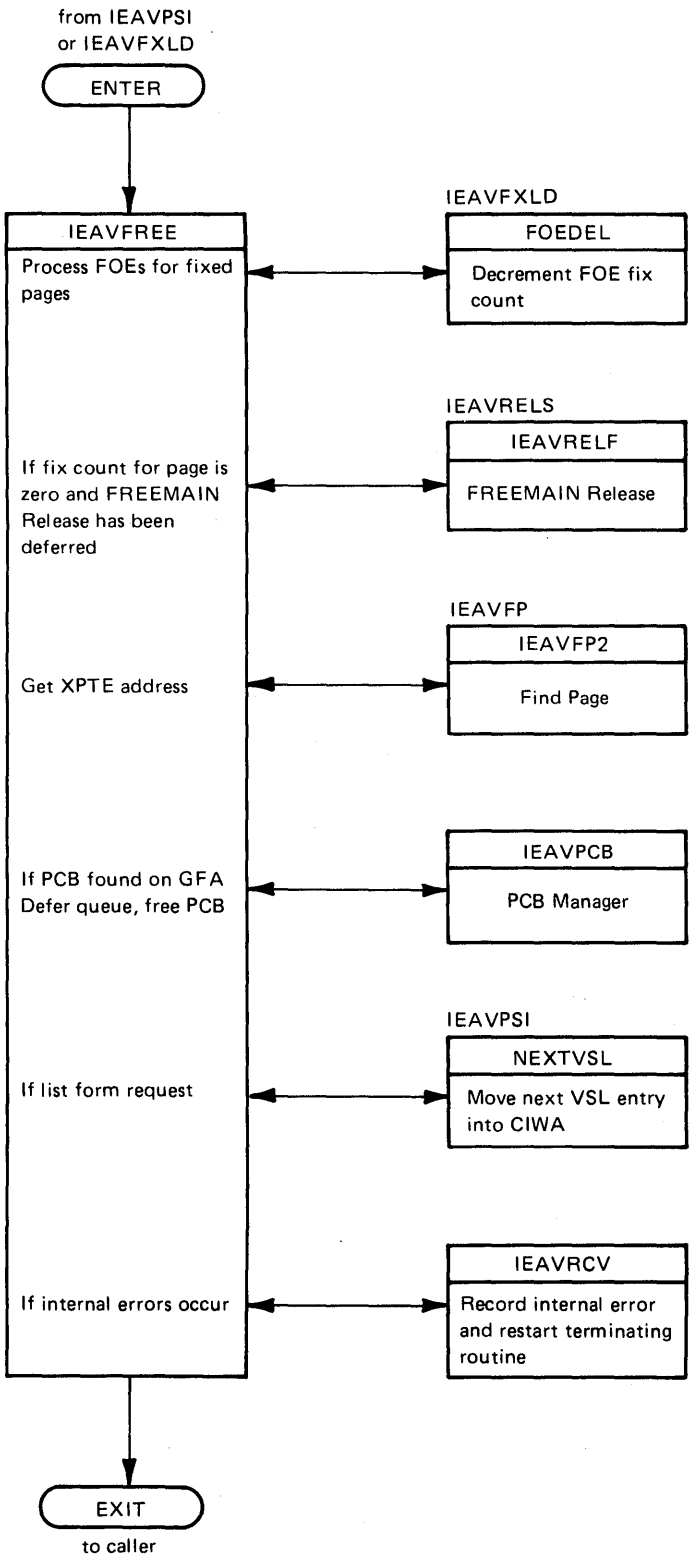


Figure 3-72. RSM Module Flow – IEAVFREE

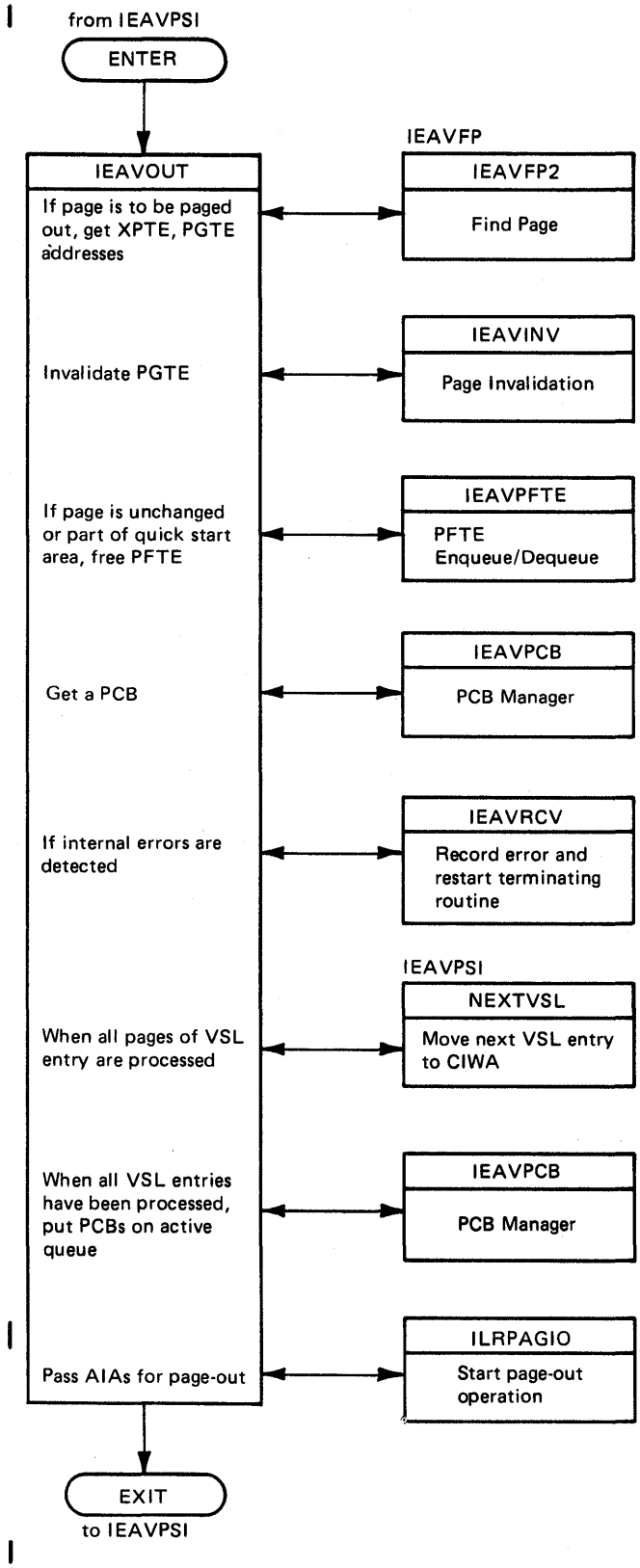


Figure 3-73. RSM Module Flow – IEAVOUT, IEAVRELS (Part 1 of 3)

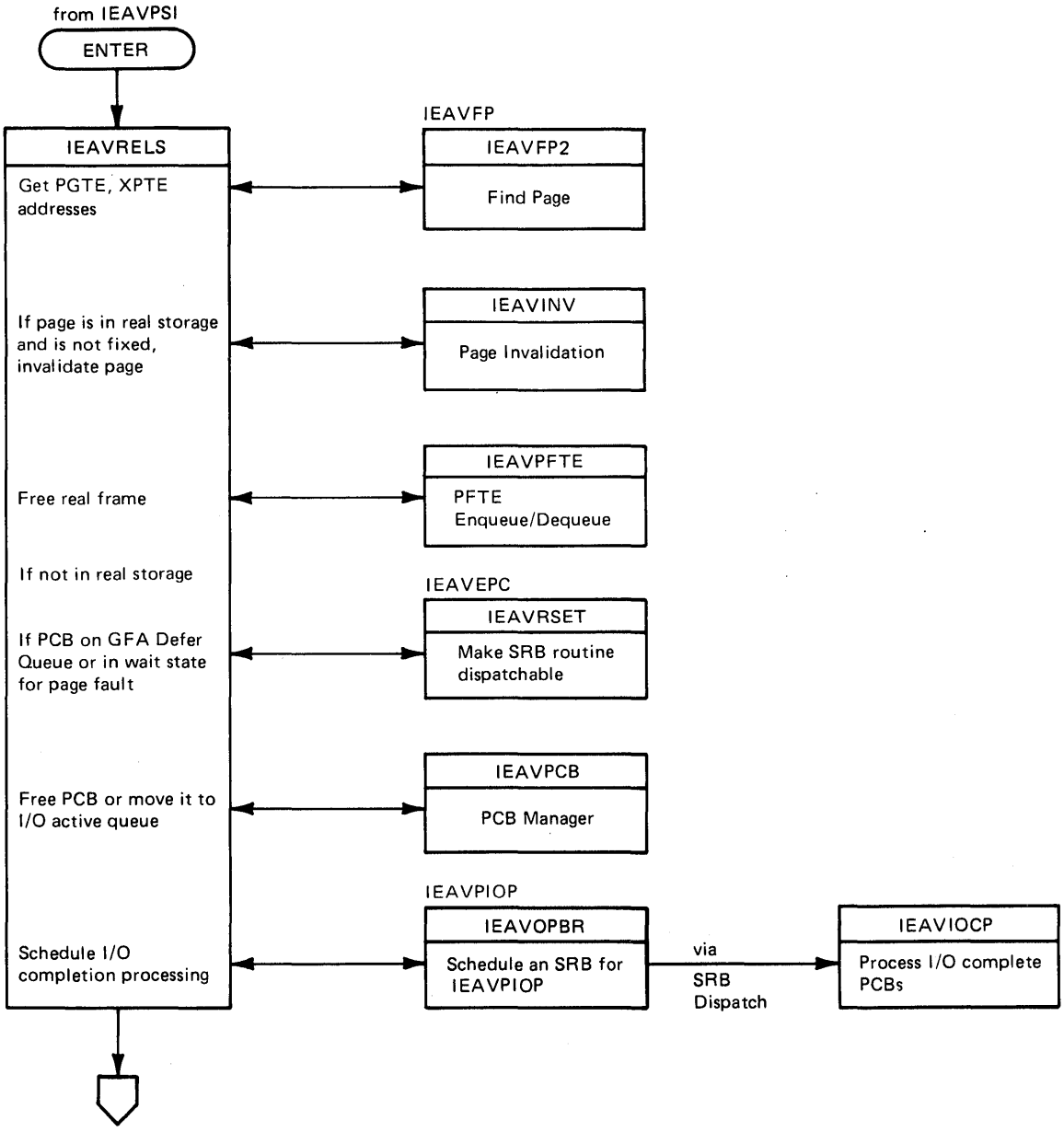


Figure 3-73. RSM Module Flow – IEAVOUT, IEAVRELS (Part 2 of 3)

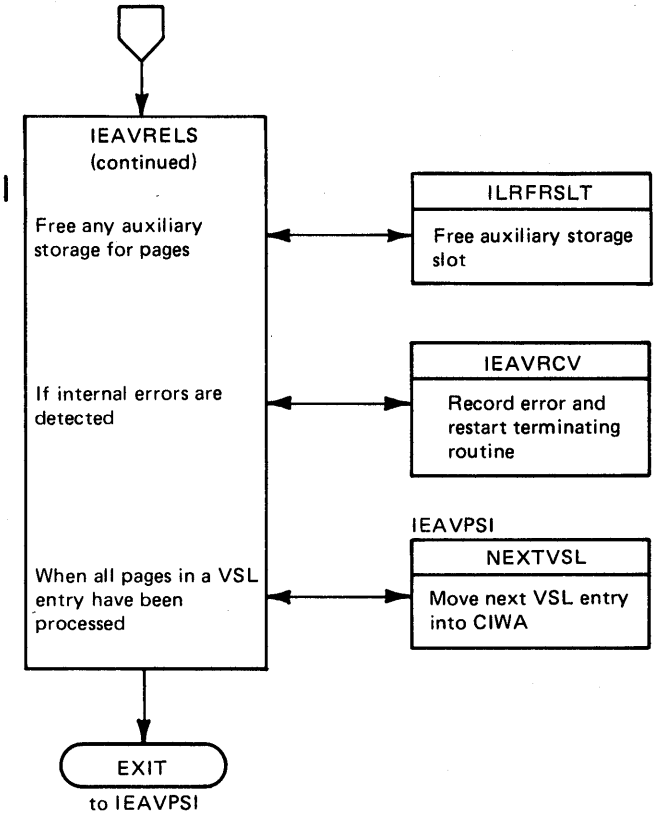


Figure 3-73. RSM Module Flow – IEAVOUT, IEAVRELS (Part 3 of 3)

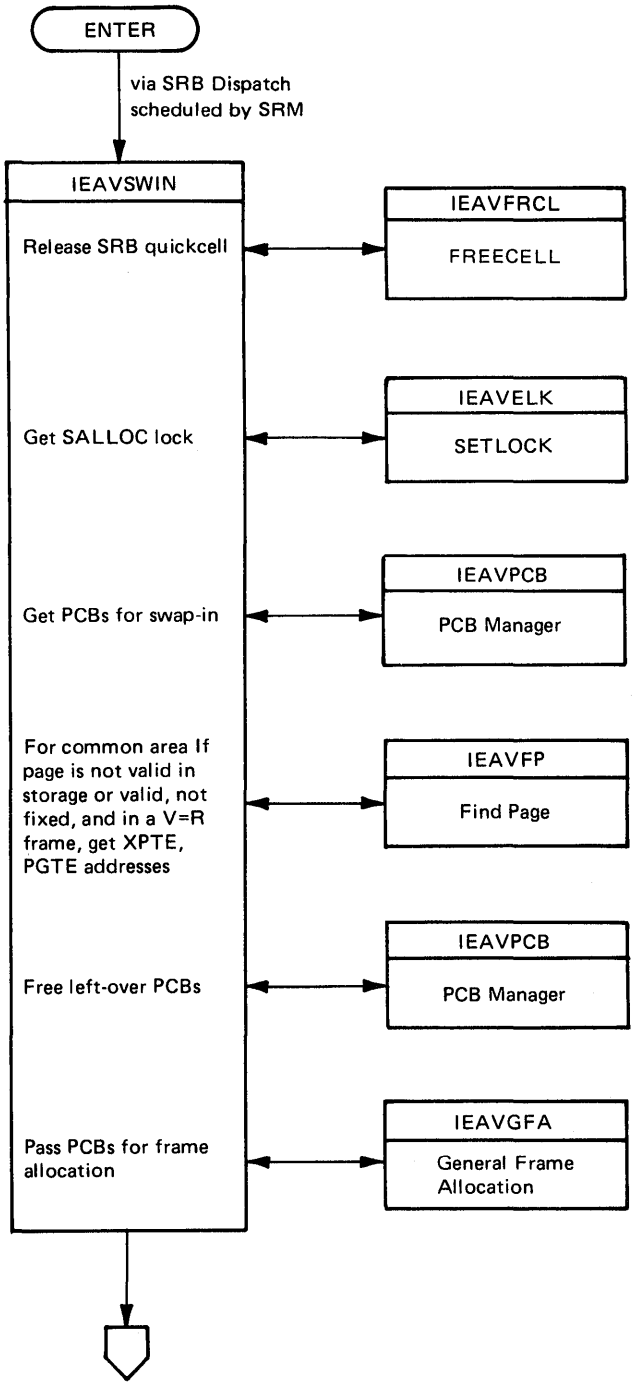


Figure 3-74. RSM Module Flow – IEAVSWIN, IEAVSWIN Root Exit, and IEAVSWIN POST Routine (Part 1 of 4)

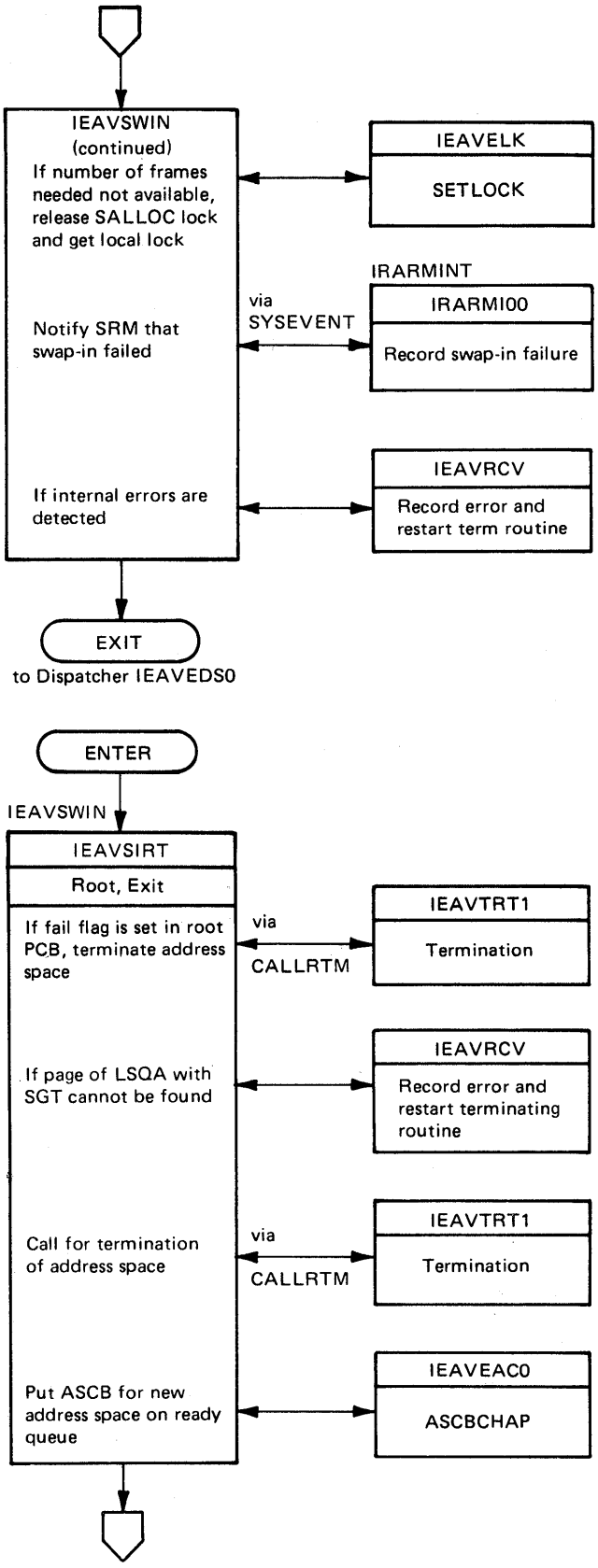


Figure 3-74. RSM Module Flow – IEAVSWIN, IEAVSWIN Root Exit, and IEAVSWIN POST Routine (Part 2 of 4)

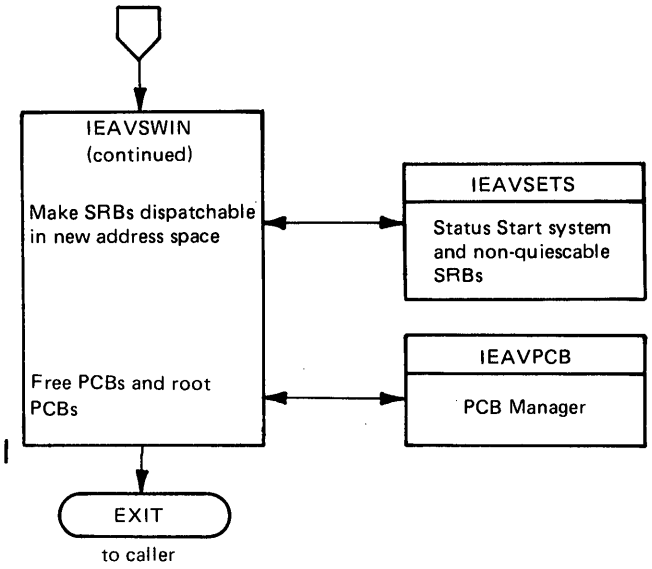


Figure 3-74. RSM Module Flow – IEAVSWIN, IEAVSWIN Root Exit, and IEAVSWIN POST Routine (Part 3 of 4)

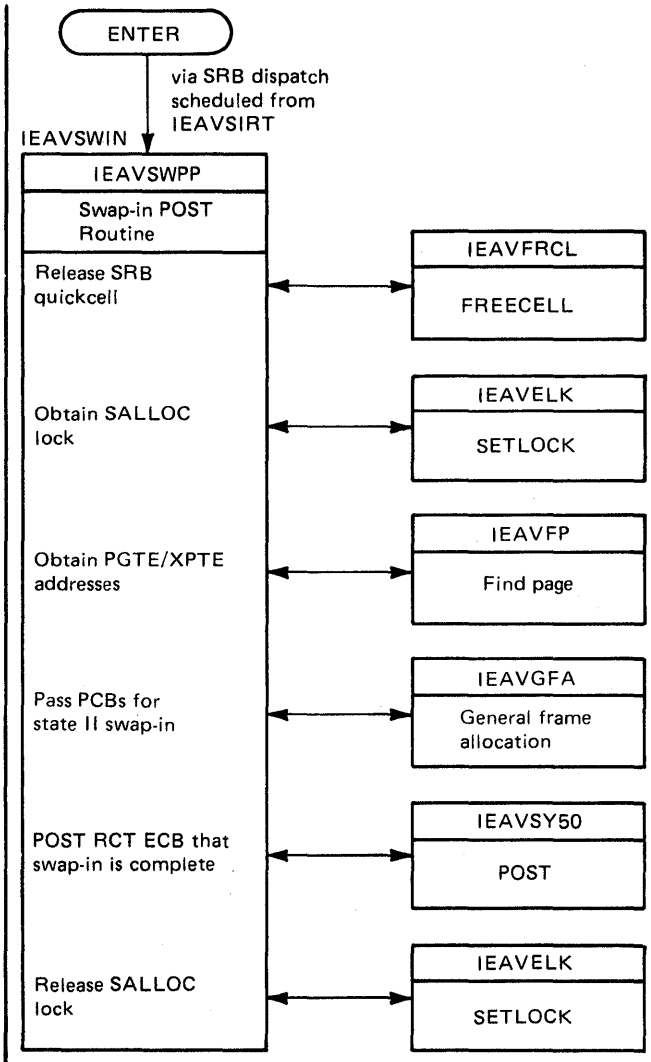


Figure 3-74. RSM Module Flow – IEAVSWIN, IEAVSWIN Root Exit, and IEAVSWIN POST Routine (Part 4 of 4)

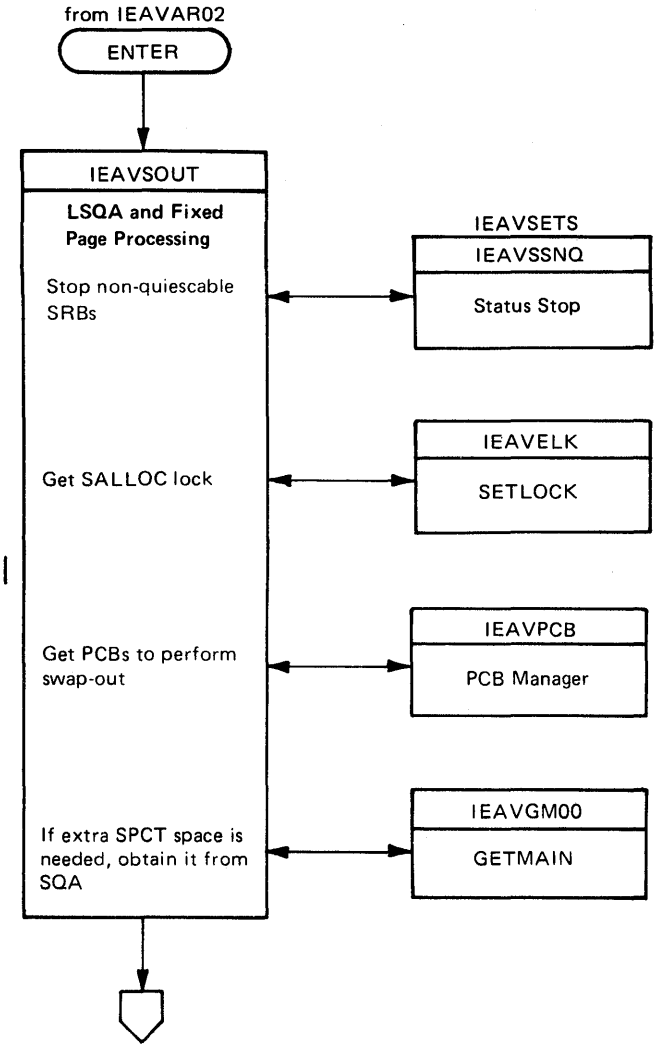


Figure 3-75. RSM Module Flow – IEAVSOUT (Part 1 of 3)

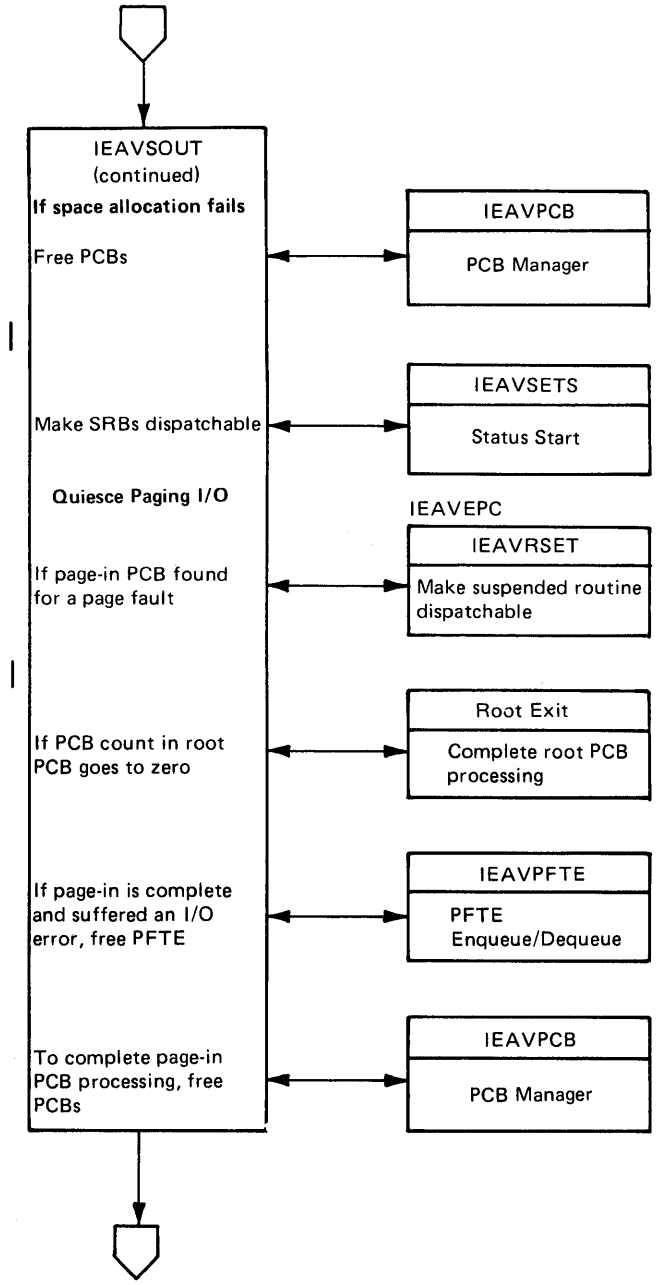


Figure 3-75. RSM Module Flow – IEAVSOUT (Part 2 of 3)

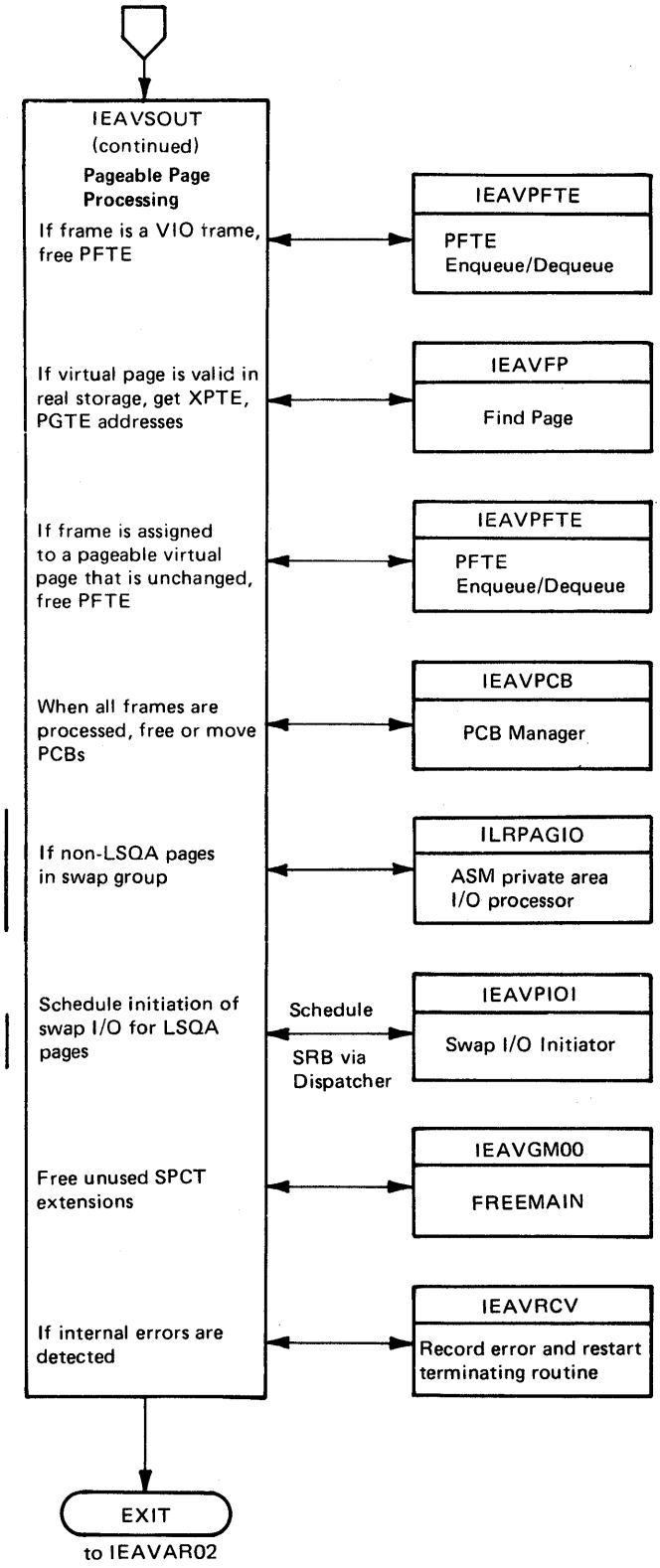


Figure 3-75. RSM Module Flow – IEAVSOUT (Part 3 of 3)

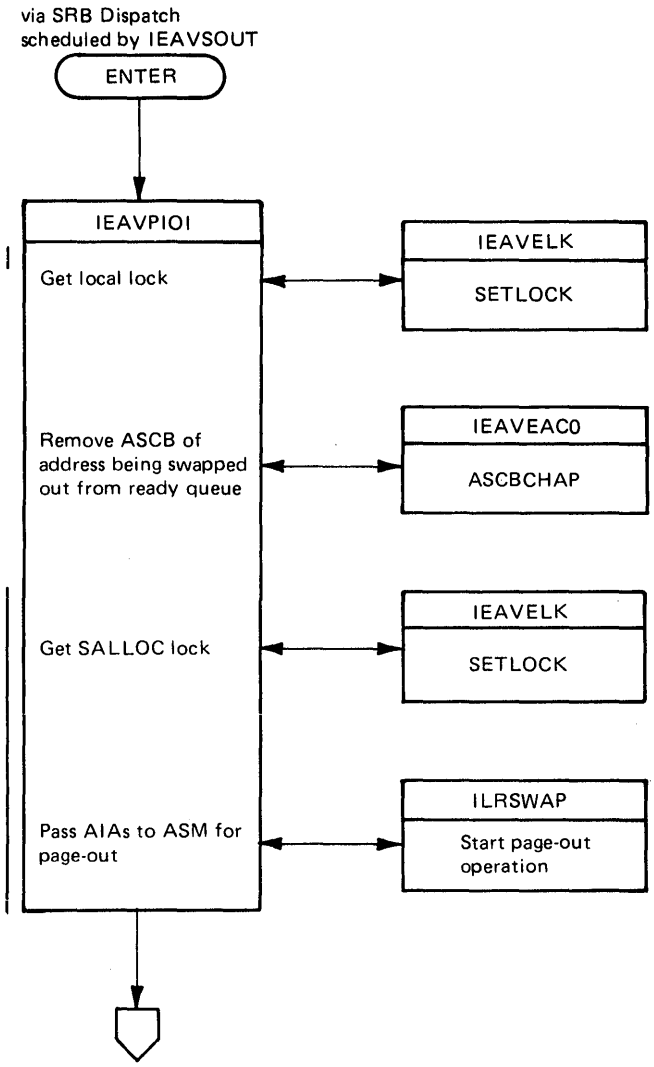


Figure 3-76. RSM Module Flow – IEAVPIOI (Part 1 of 2)

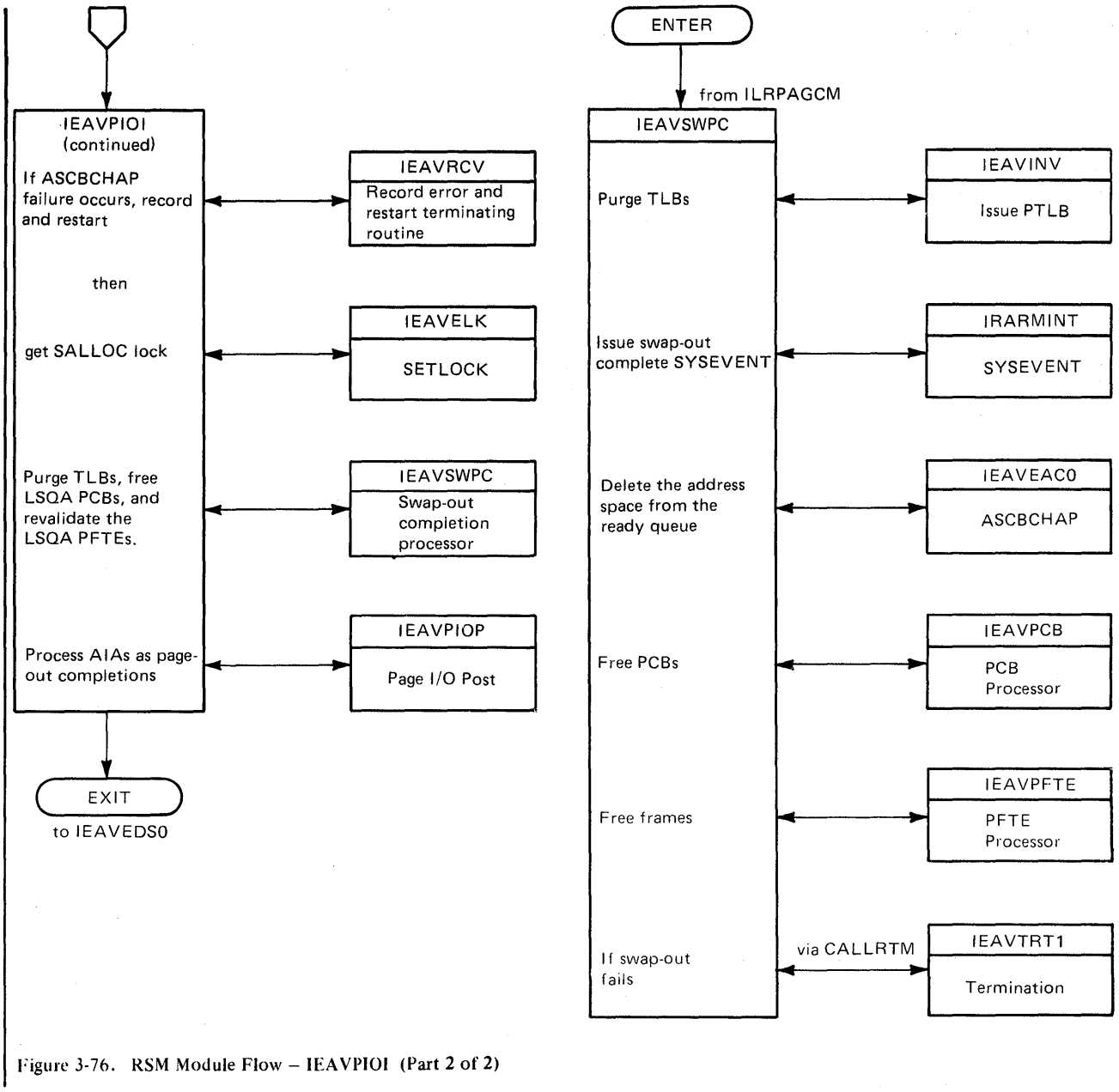


Figure 3-76. RSM Module Flow – IEAVPIOI (Part 2 of 2)

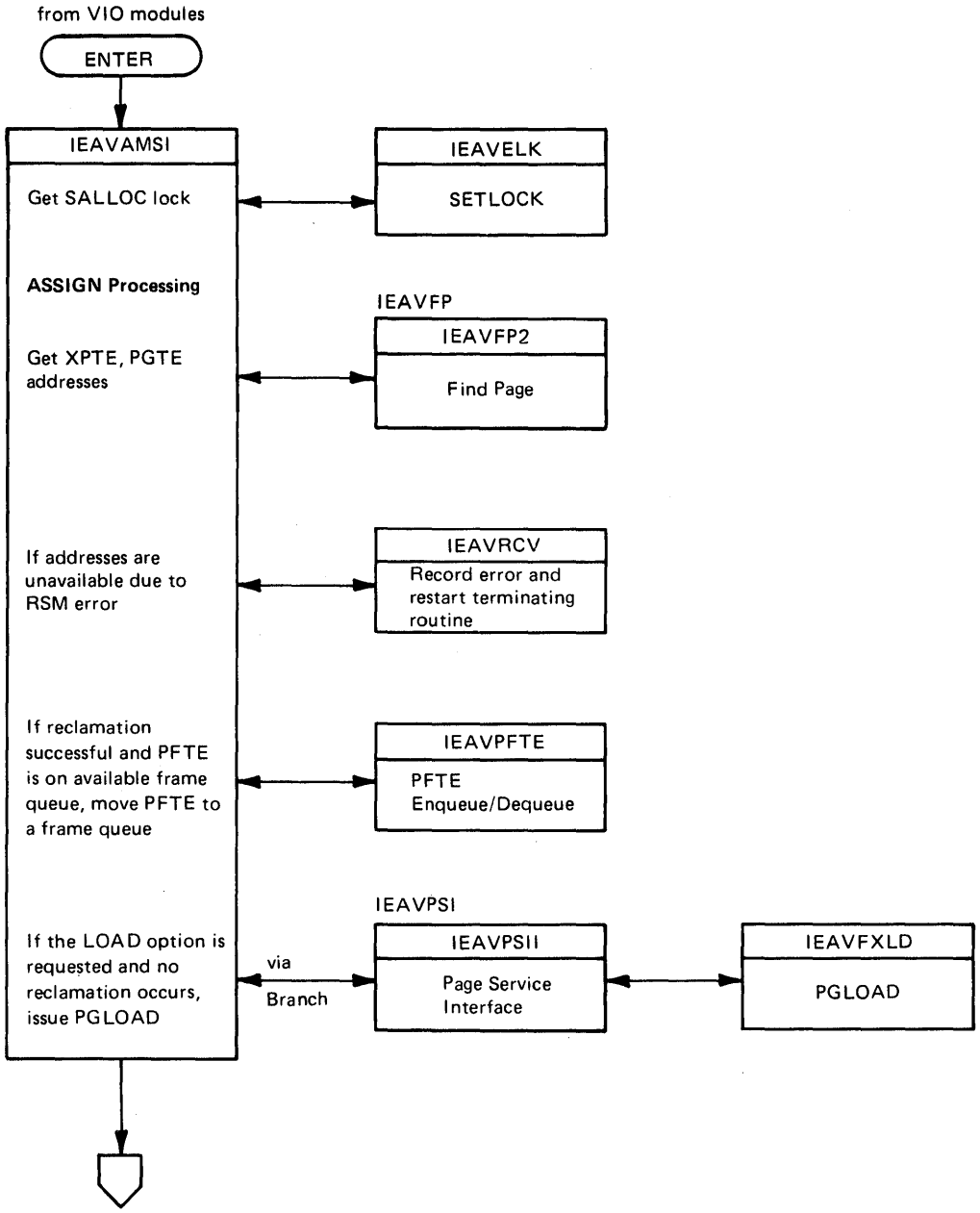


Figure 3-77. RSM Module Flow – IEAVAMSI (Part 1 of 4)

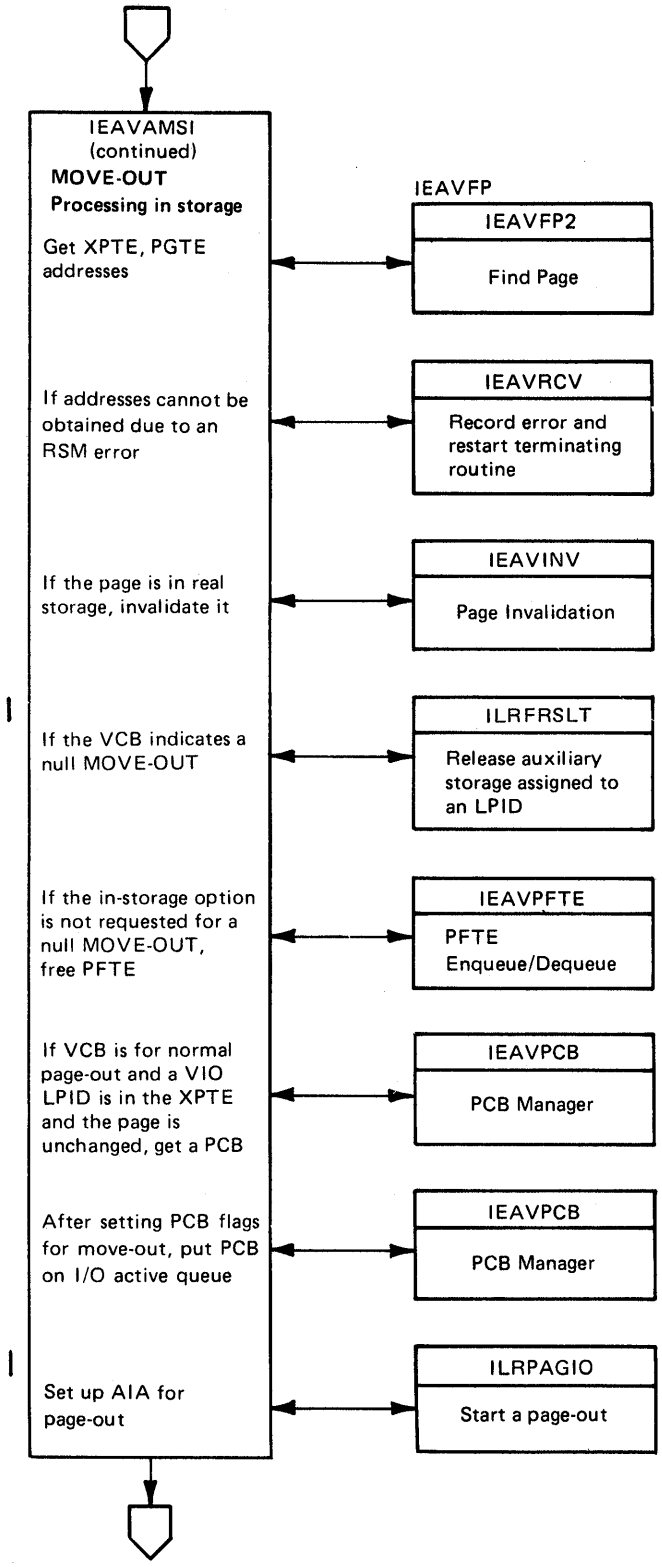


Figure 3-77. RSM Module Flow – IEAVAMSI (Part 2 of 4)

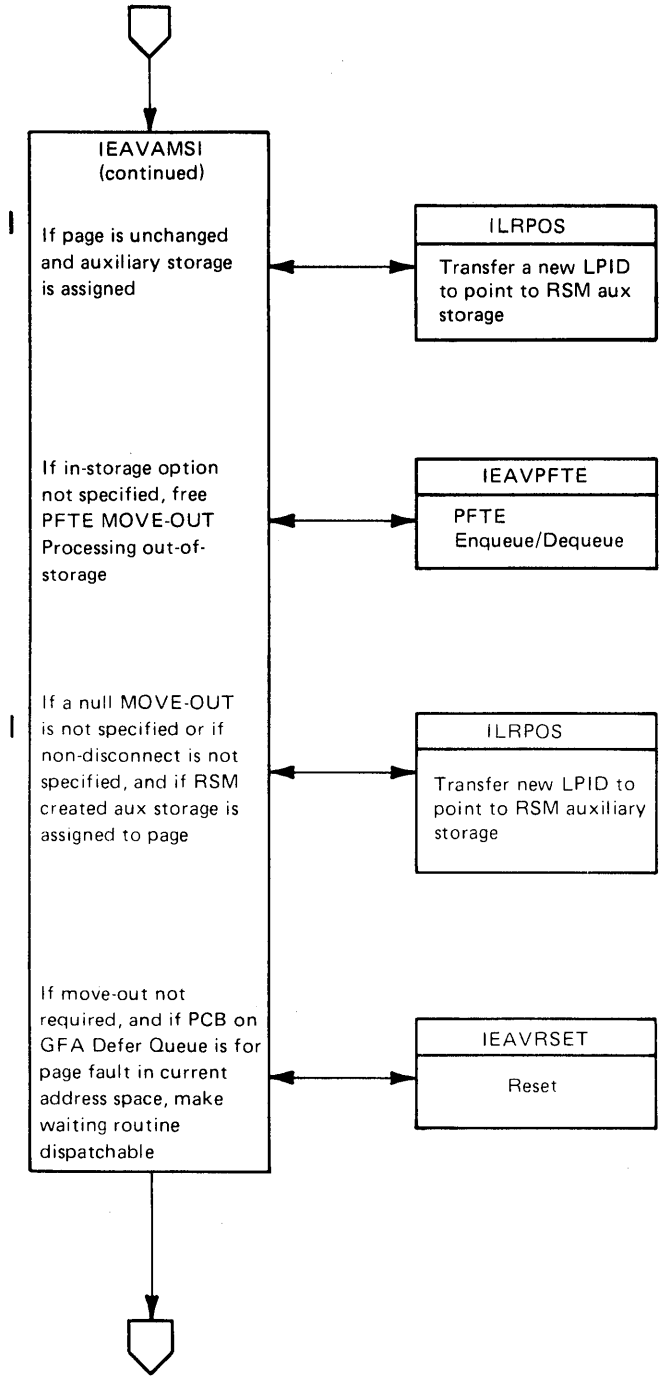


Figure 3-77. RSM Module Flow – IEAVAMSI (Part 3 of 4)

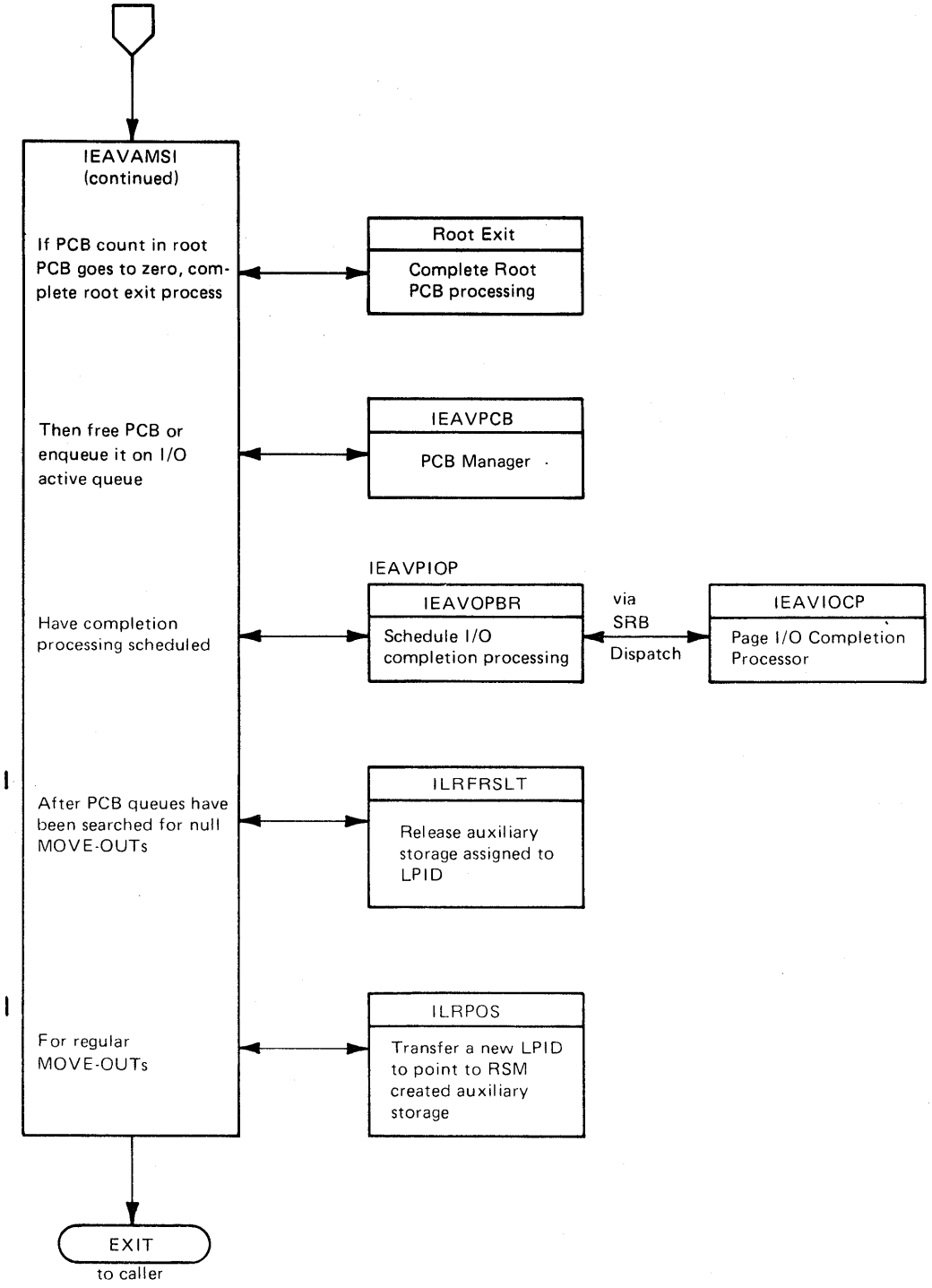


Figure 3-77. RSM Module Flow - IEAVAMSI (Part 4 of 4)

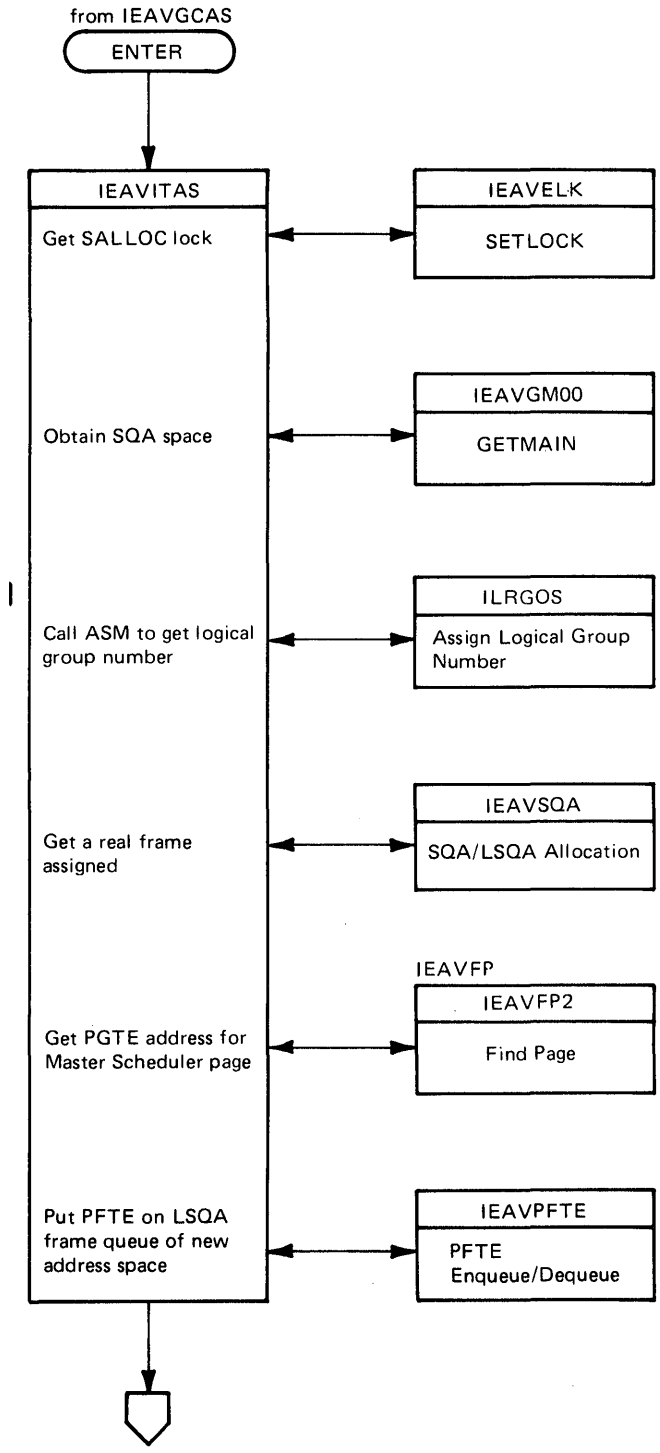


Figure 3-78. RSM Module Flow – IEAVITAS (Part 1 of 2)

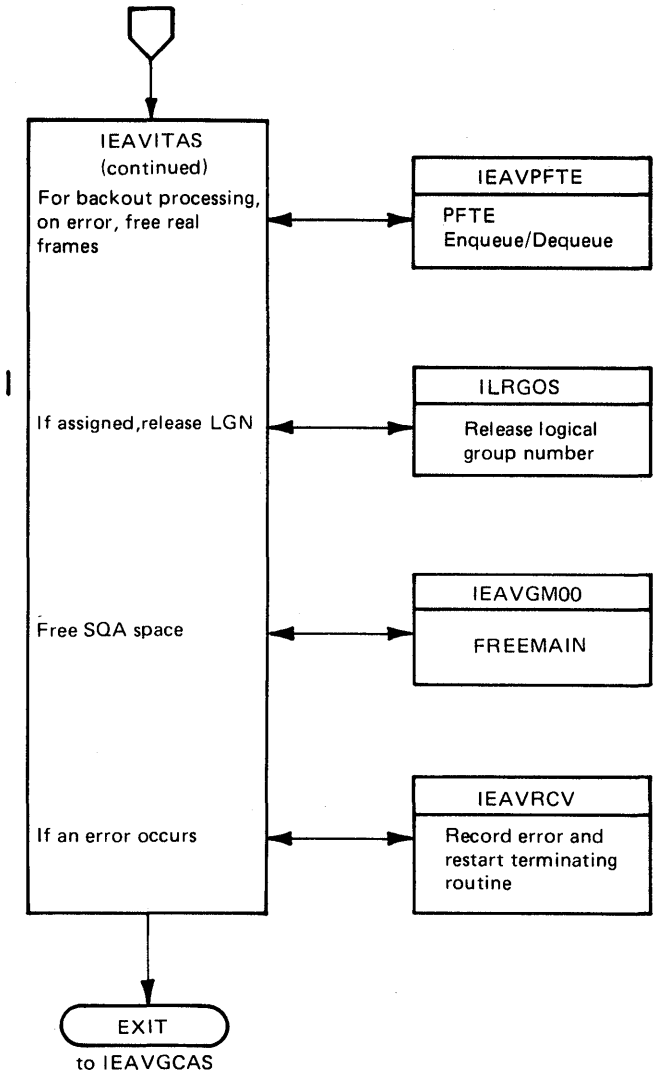


Figure 3-78. RSM Module Flow – IEAVITAS (Part 2 of 2)

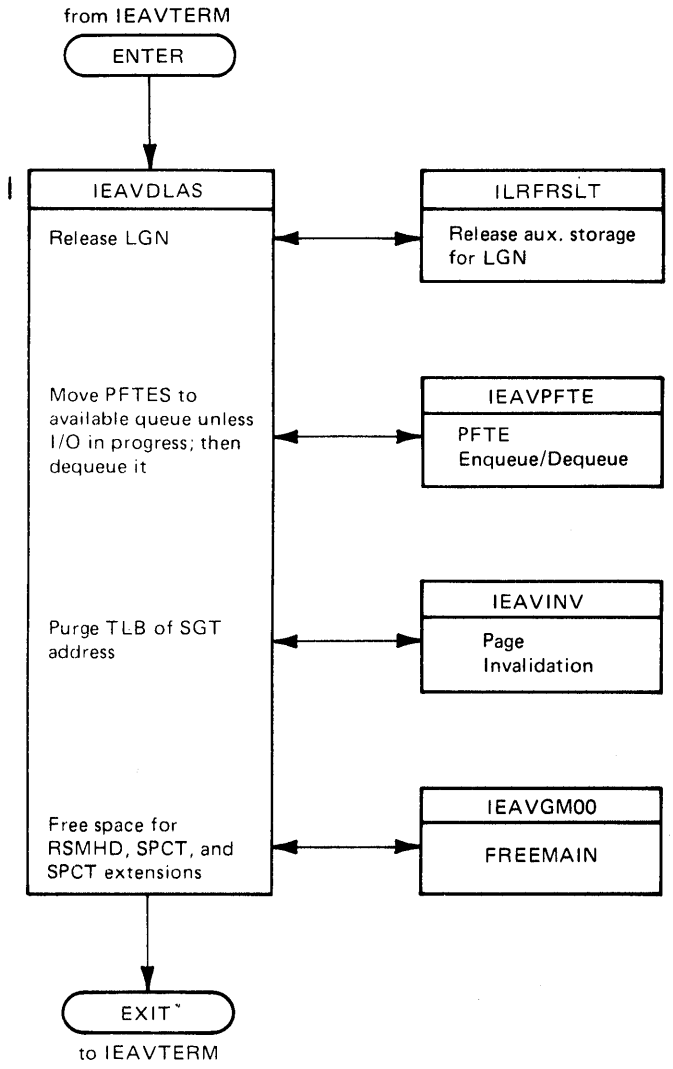


Figure 3-79. RSM Module Flow – IEAVDLAS (Part 1 of 2)

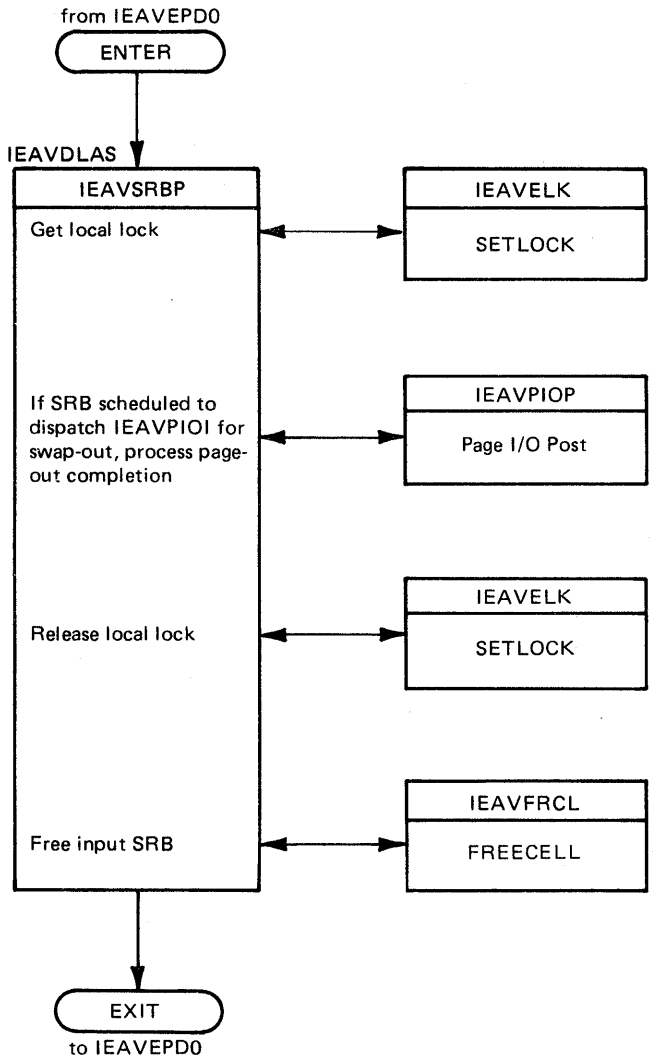


Figure 3-79. RSM Module Flow – IEAVDLAS (Part 2 of 2)

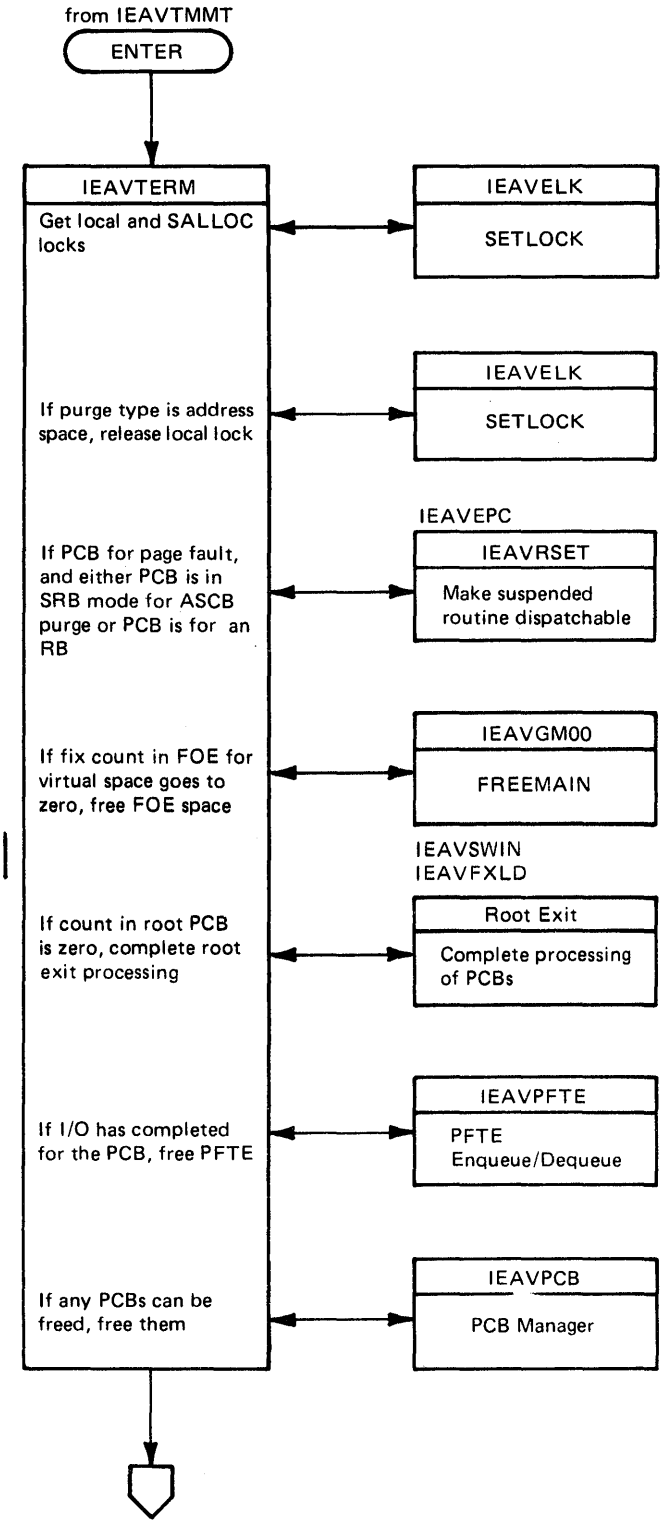


Figure 3-80. RSM Module Flow – IEAVTERM (Part 1 of 2)

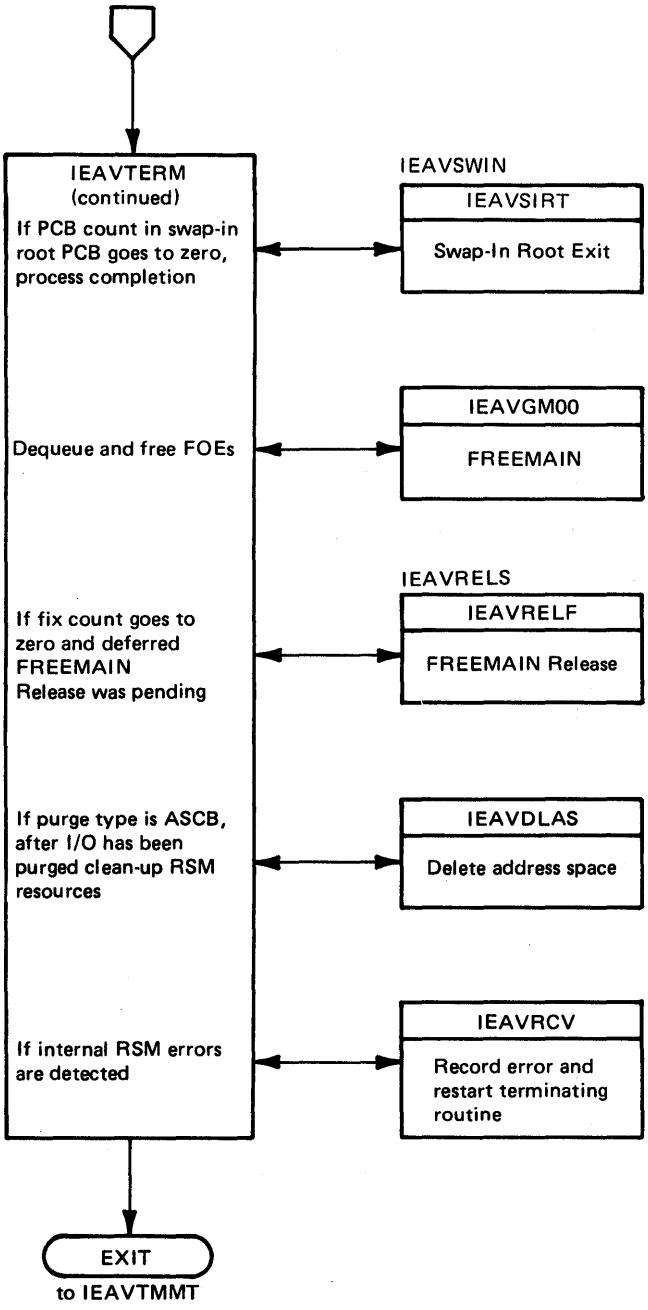


Figure 3-80. RSM Module Flow – IEAVTERM (Part 2 of 2)

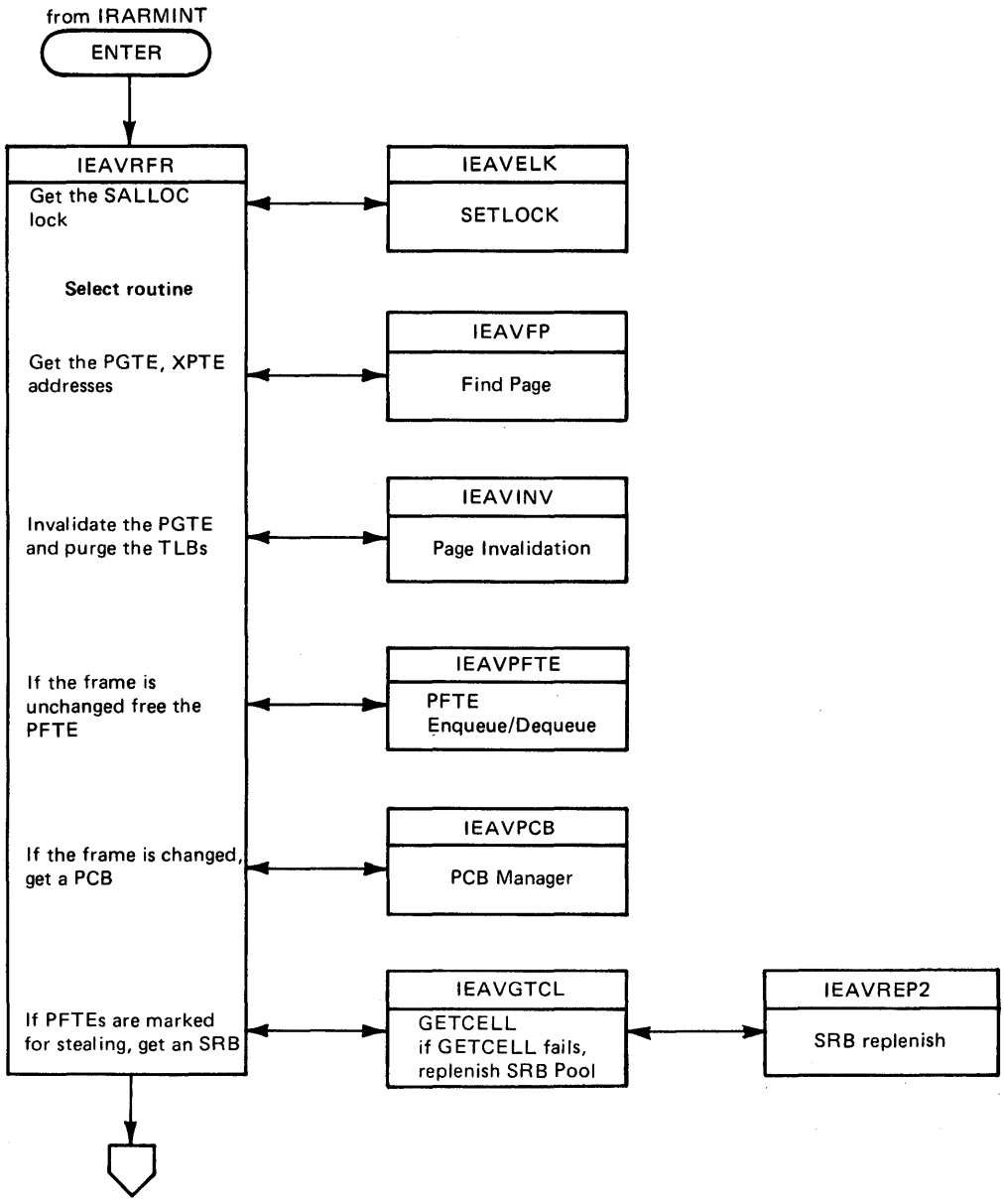


Figure 3-81. RSM Module Flow – IEAVRFR (Part 1 of 3)

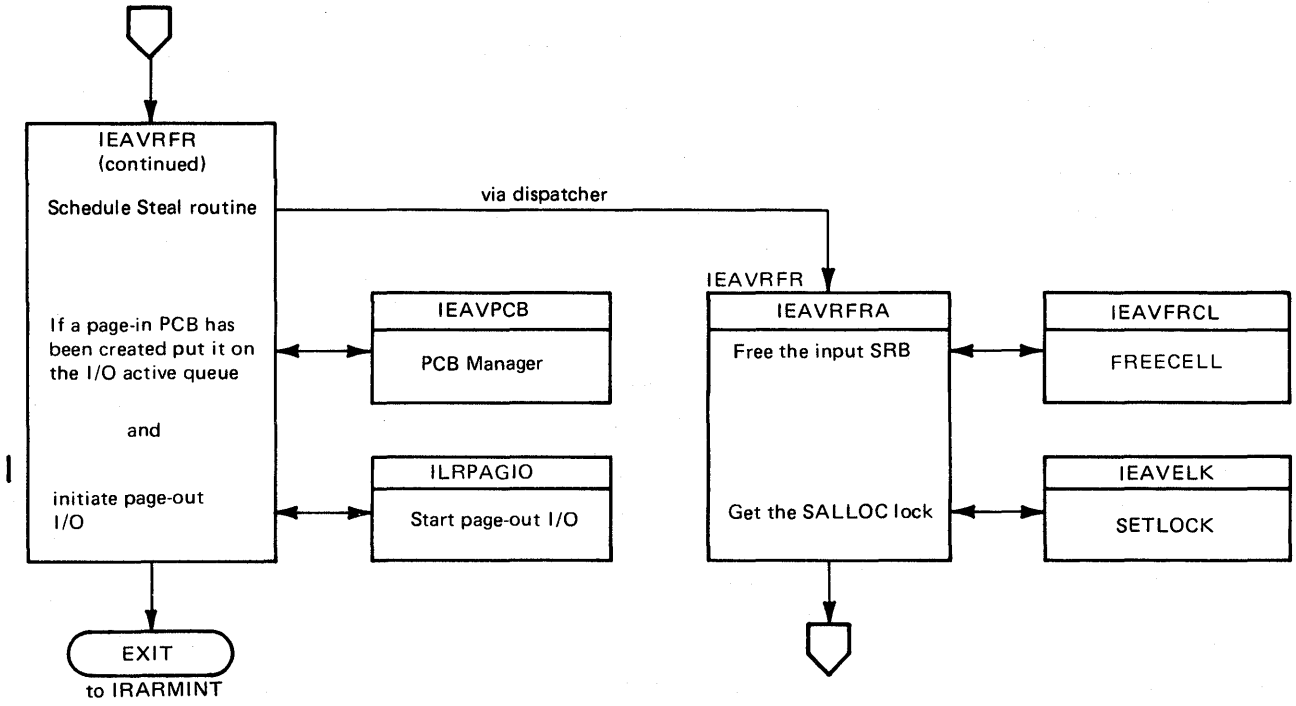


Figure 3-81. RSM Module Flow – IEAVRFR (Part 2 of 3)

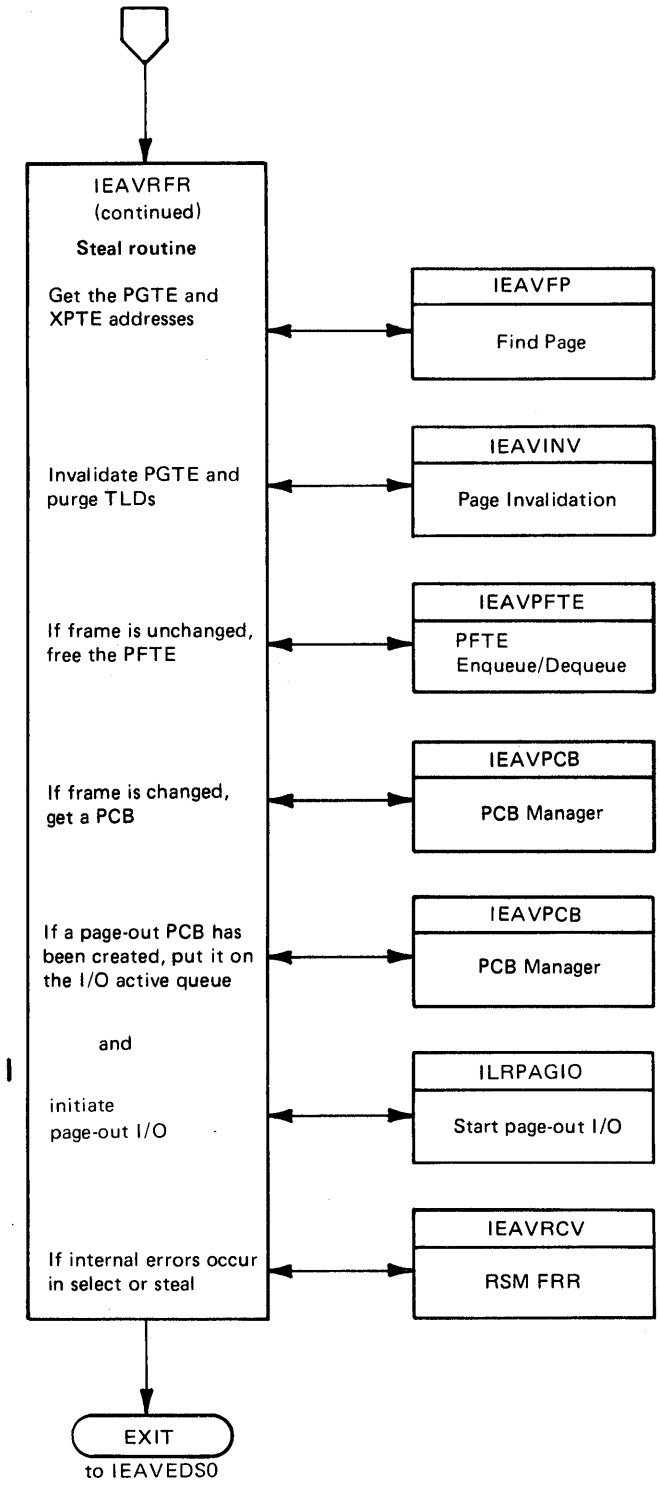


Figure 3-81. RSM Module Flow – IEAVRFR (Part 3 of 3)

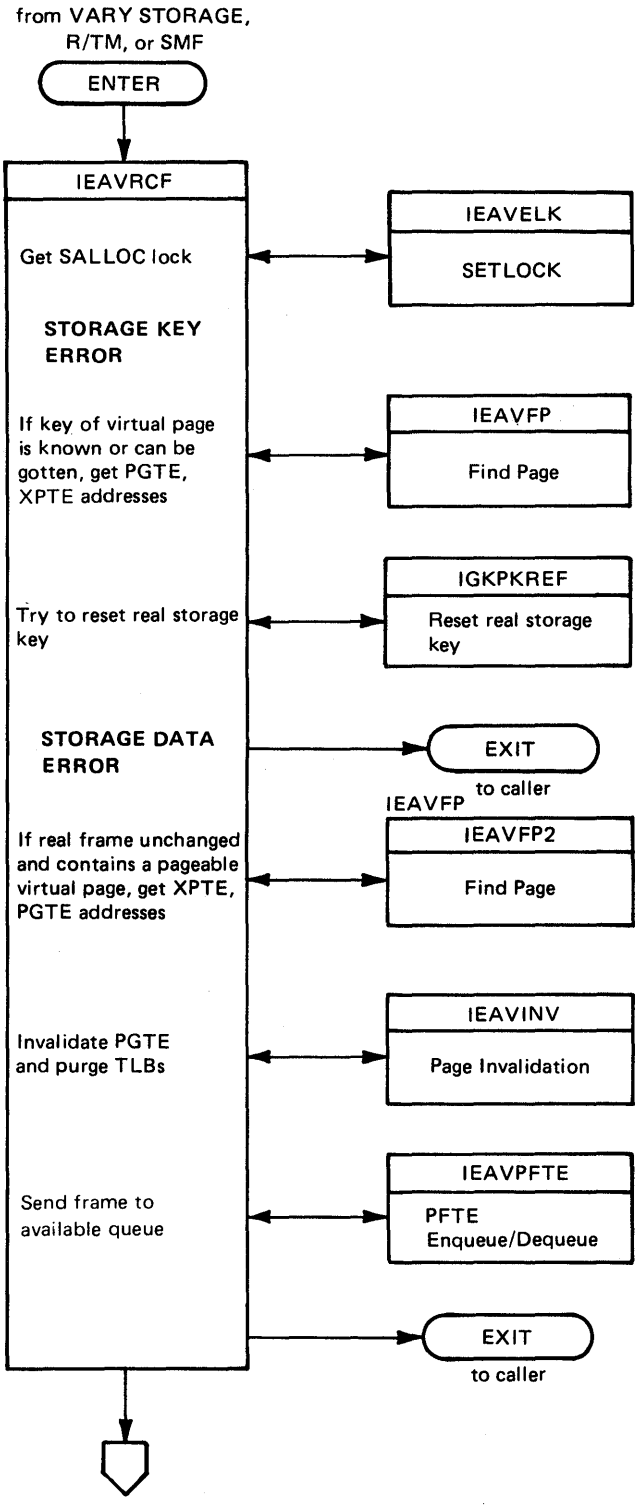


Figure 3-82. RSM Module Flow – IEAVRCF (Part 1 of 2)

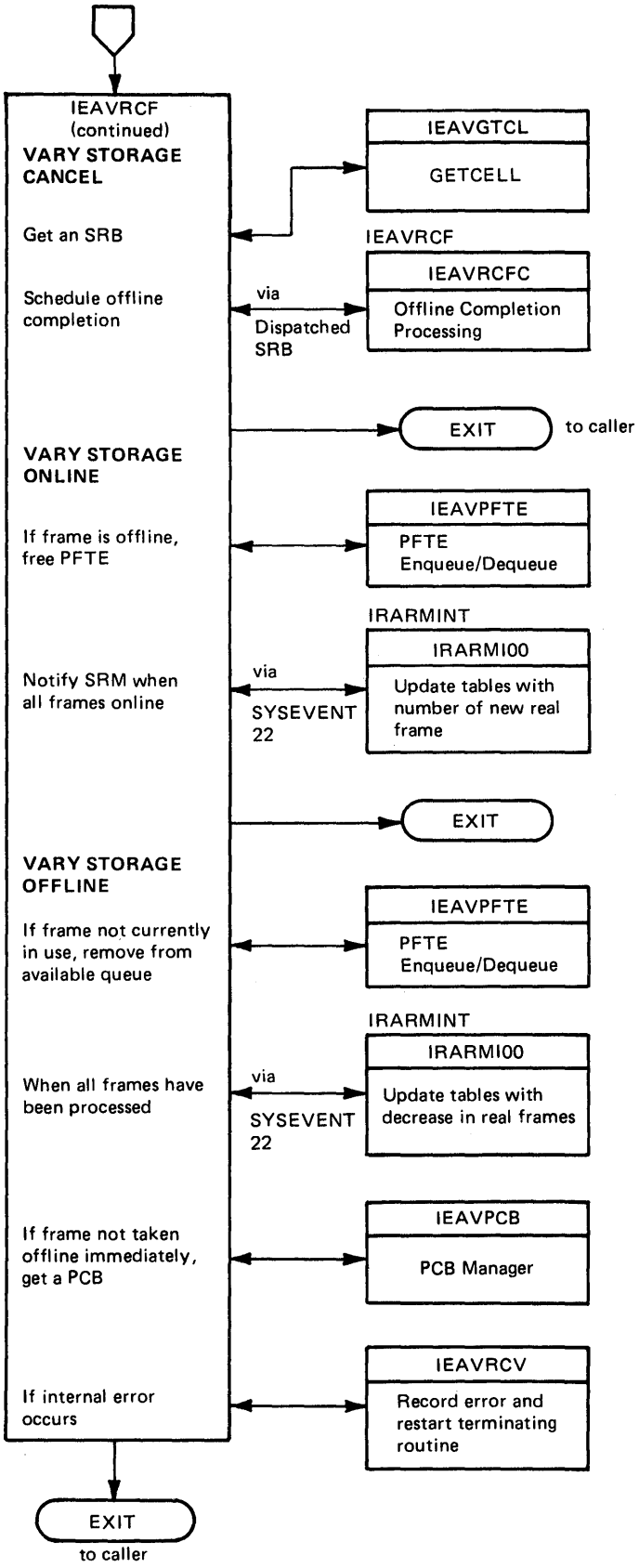


Figure 3-82. RSM Module Flow – IEAVRCF (Part 2 of 2)

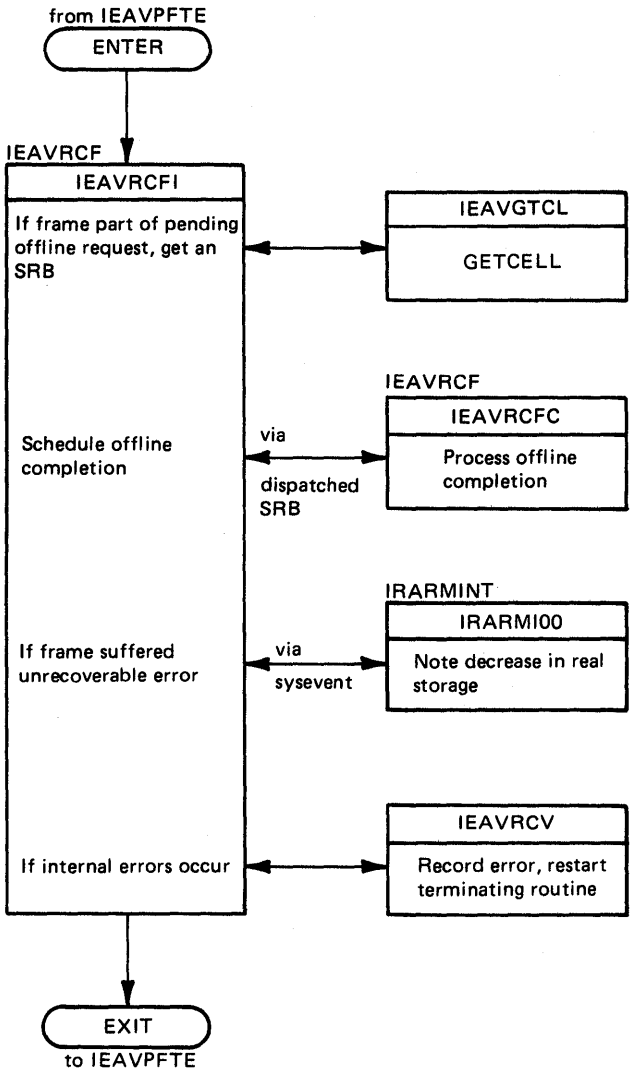


Figure 3-83. RSM Module Flow – IEAVRCF Offline Interception and Offline Completion (Part 1 of 2)

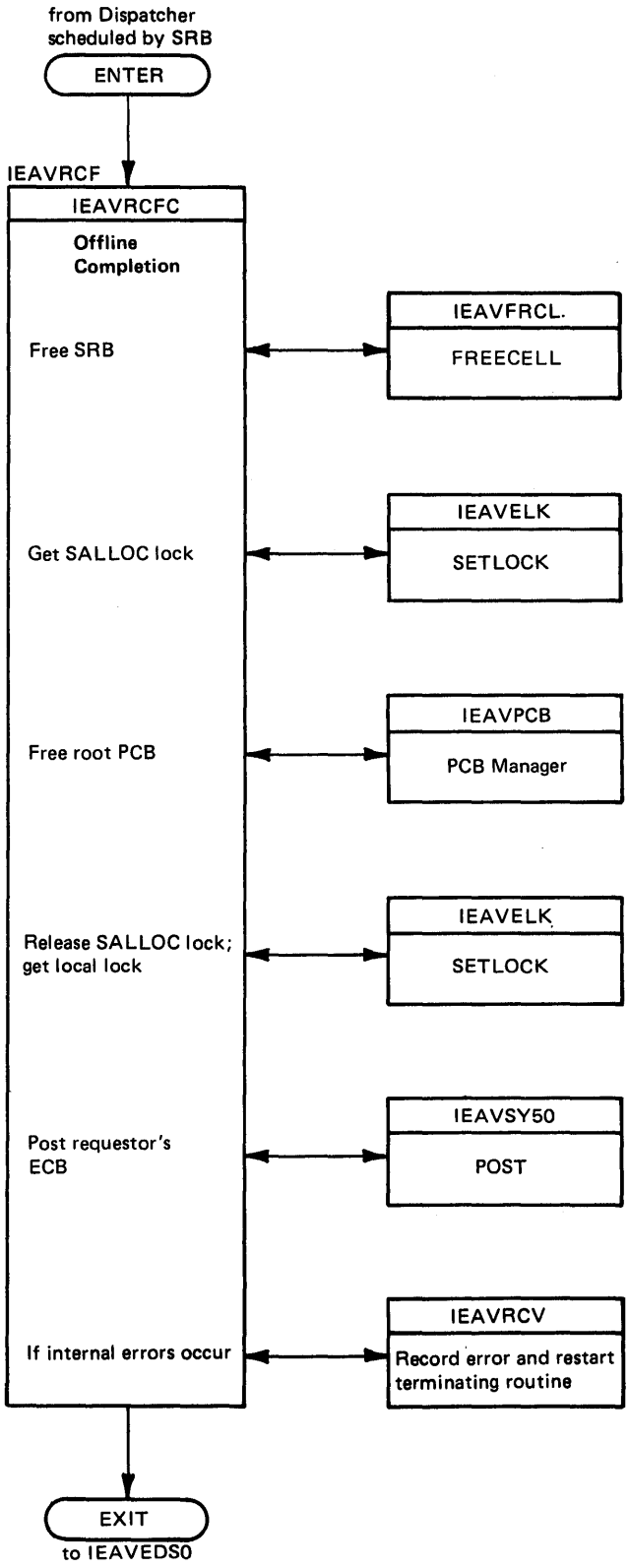


Figure 3-83. RSM Module Flow – IEAVRCF Offline Interception and Offline Completion (Part 2 of 2)

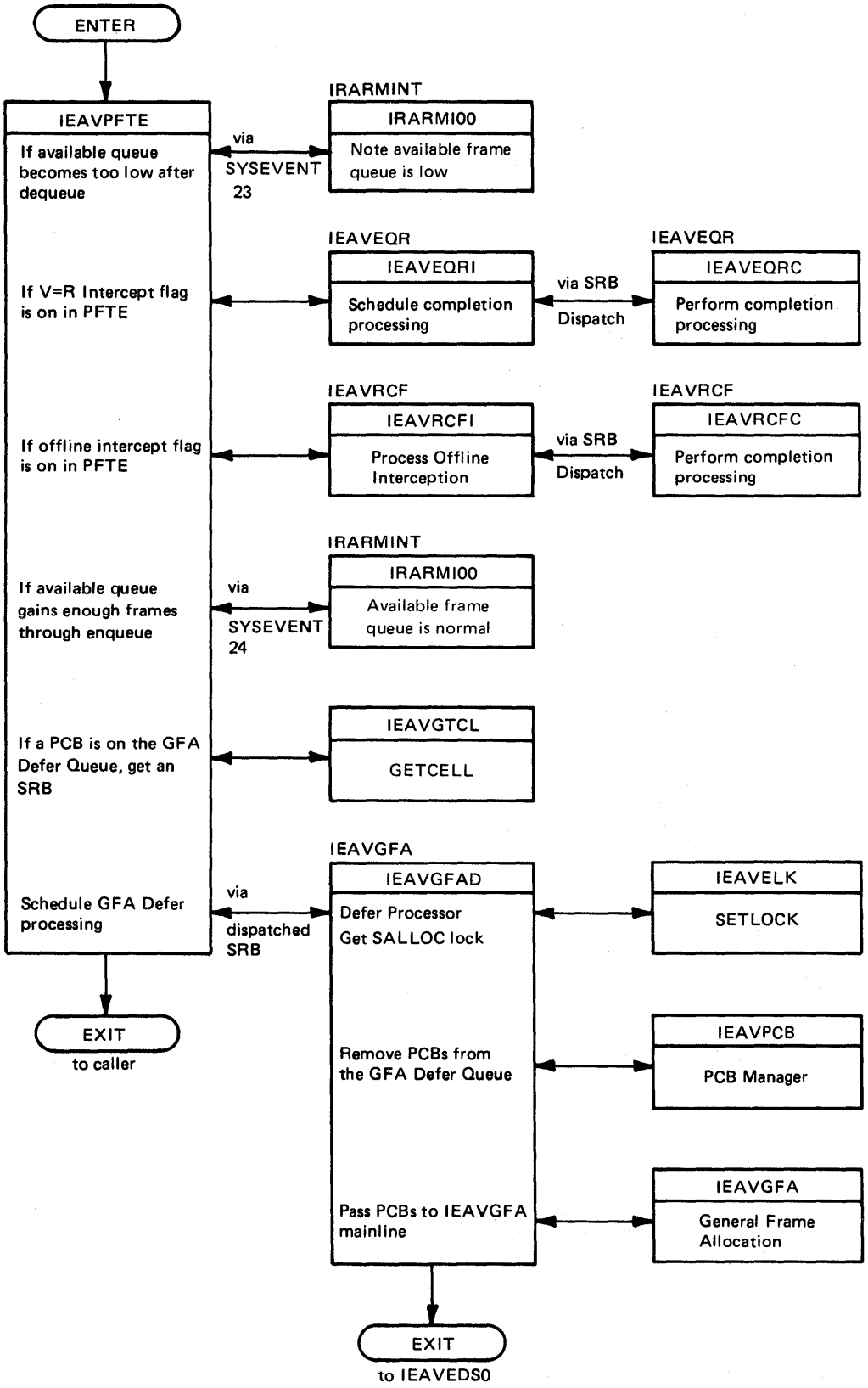


Figure 3-84. RSM Module Flow – IEAVPFTE

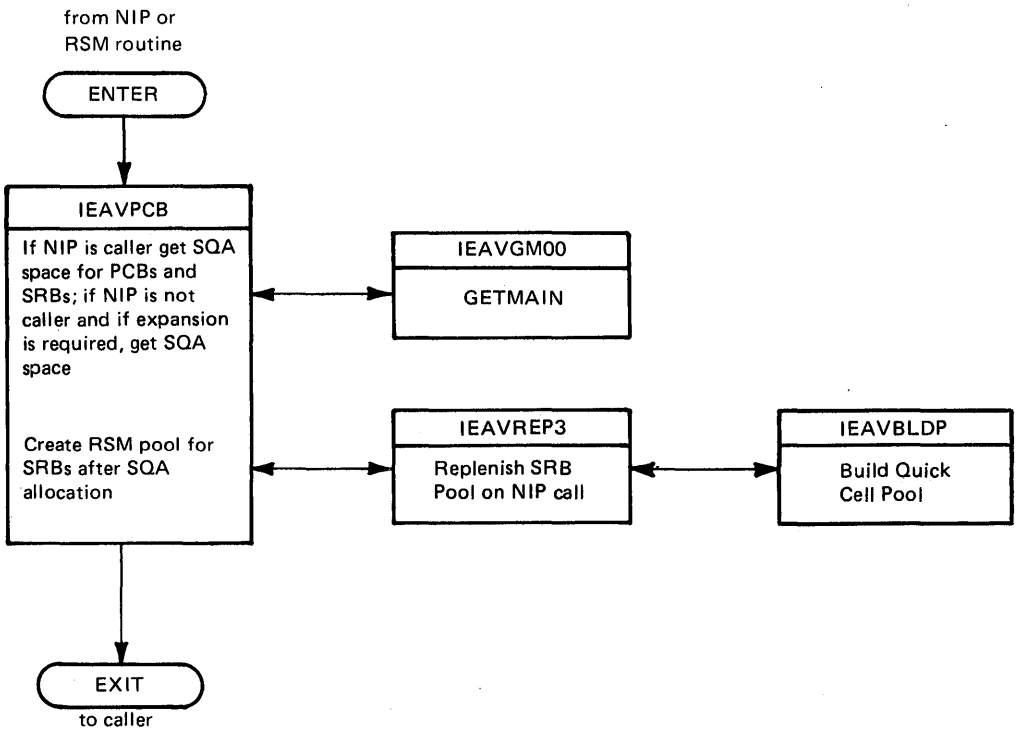


Figure 3-85. RSM Module Flow – IEAVPCB

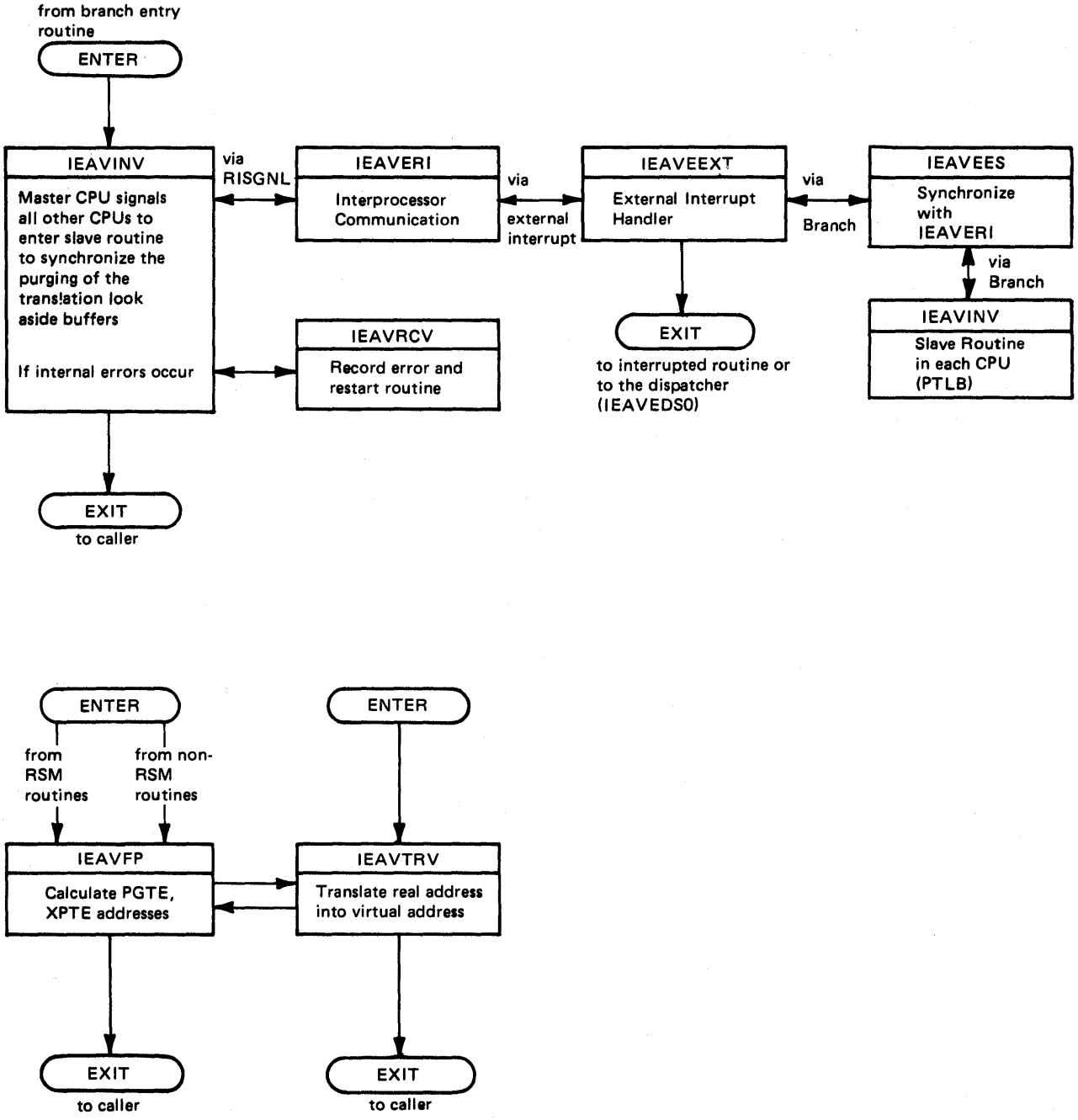


Figure 3-86. RSM Module Flow – IEAVINV, IEAVFP, IEAVTRV

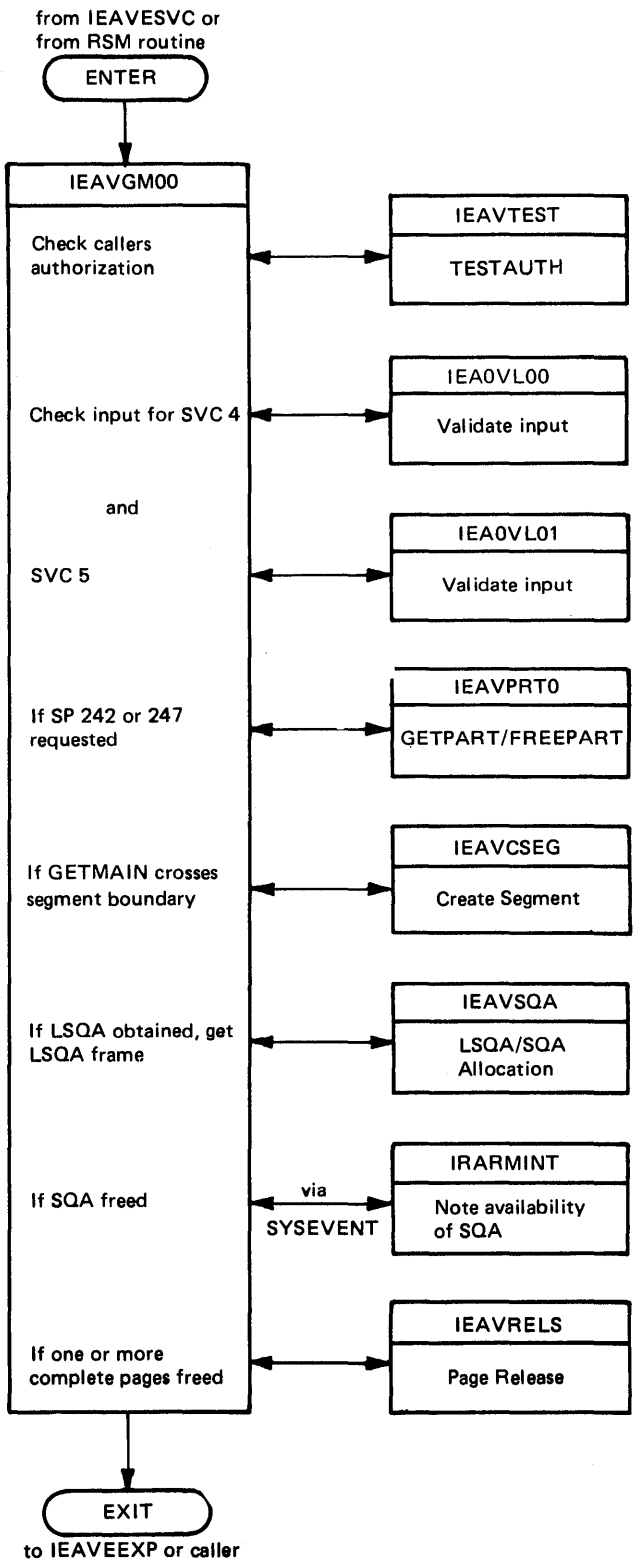


Figure 3-87. VSM Module Flow (Part 1 of 4)

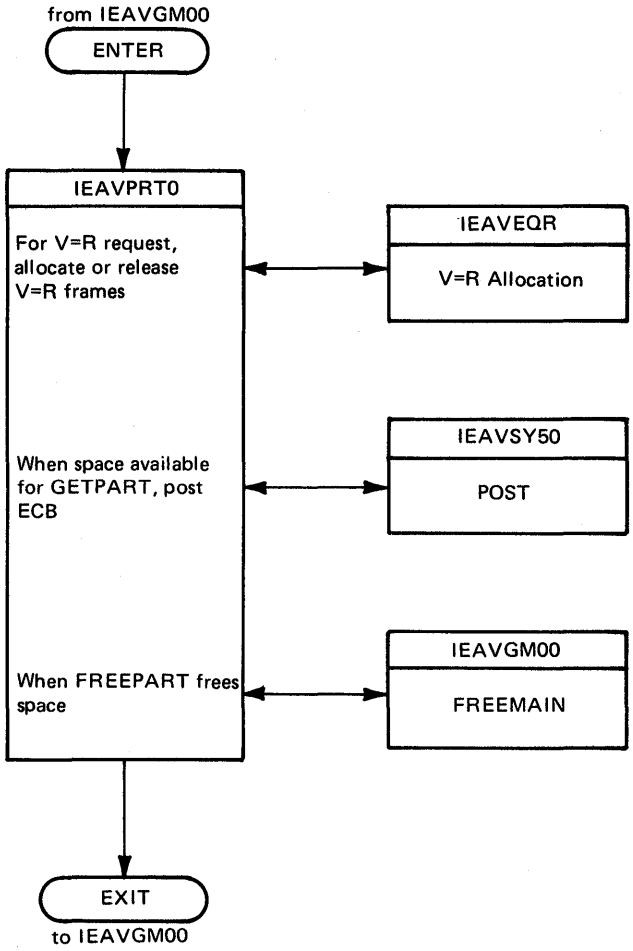
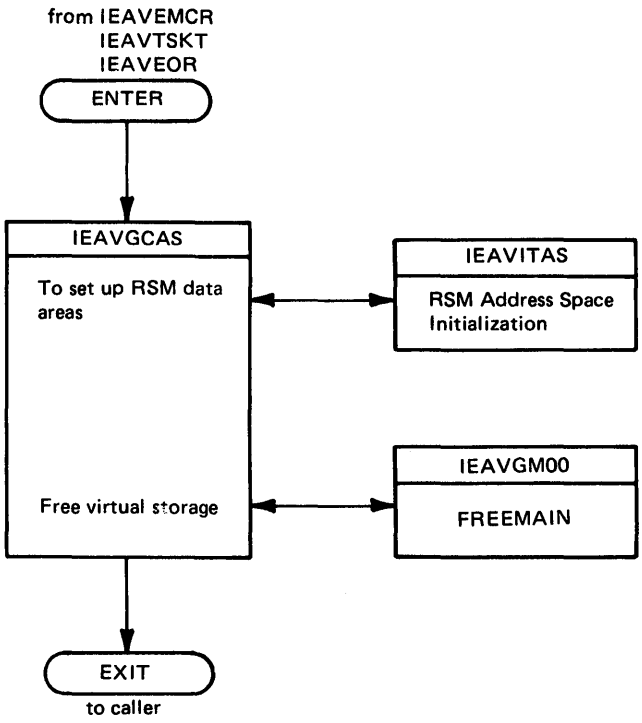


Figure 3-87. VSM Module Flow (Part 2 of 4)



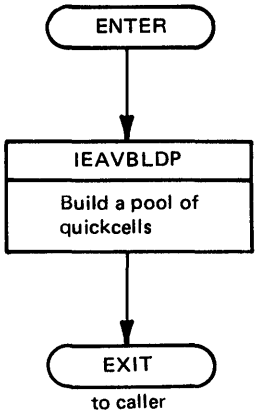
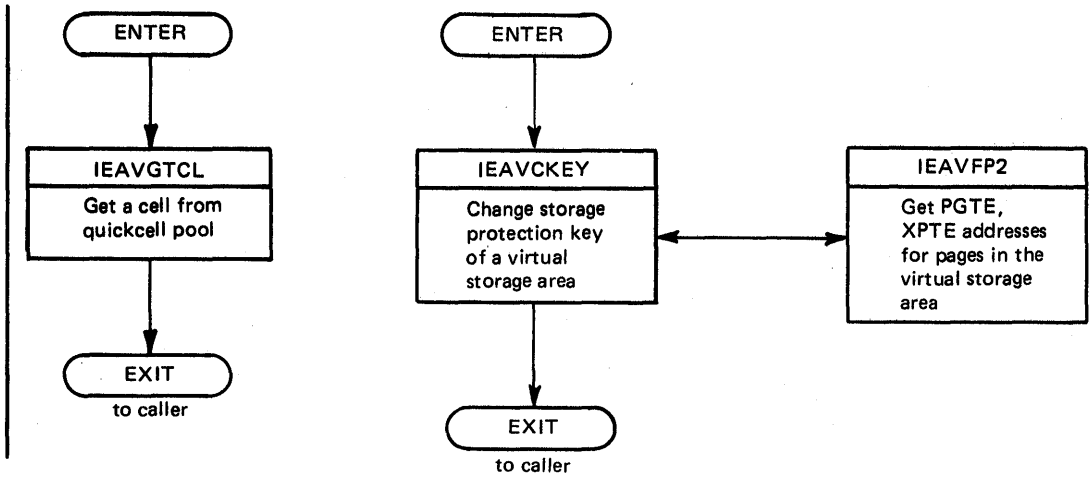
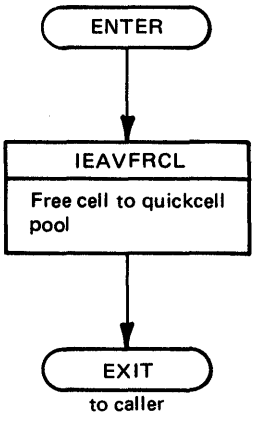


Figure 3-87. VSM Module Flow (Part 3 of 4)





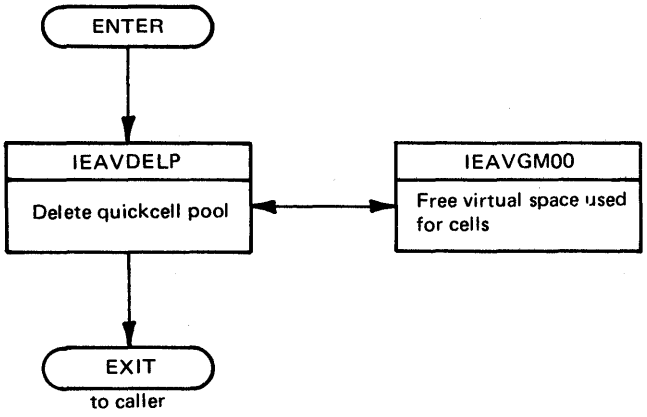


Figure 3-87. VSM Module Flow (Part 4 of 4)

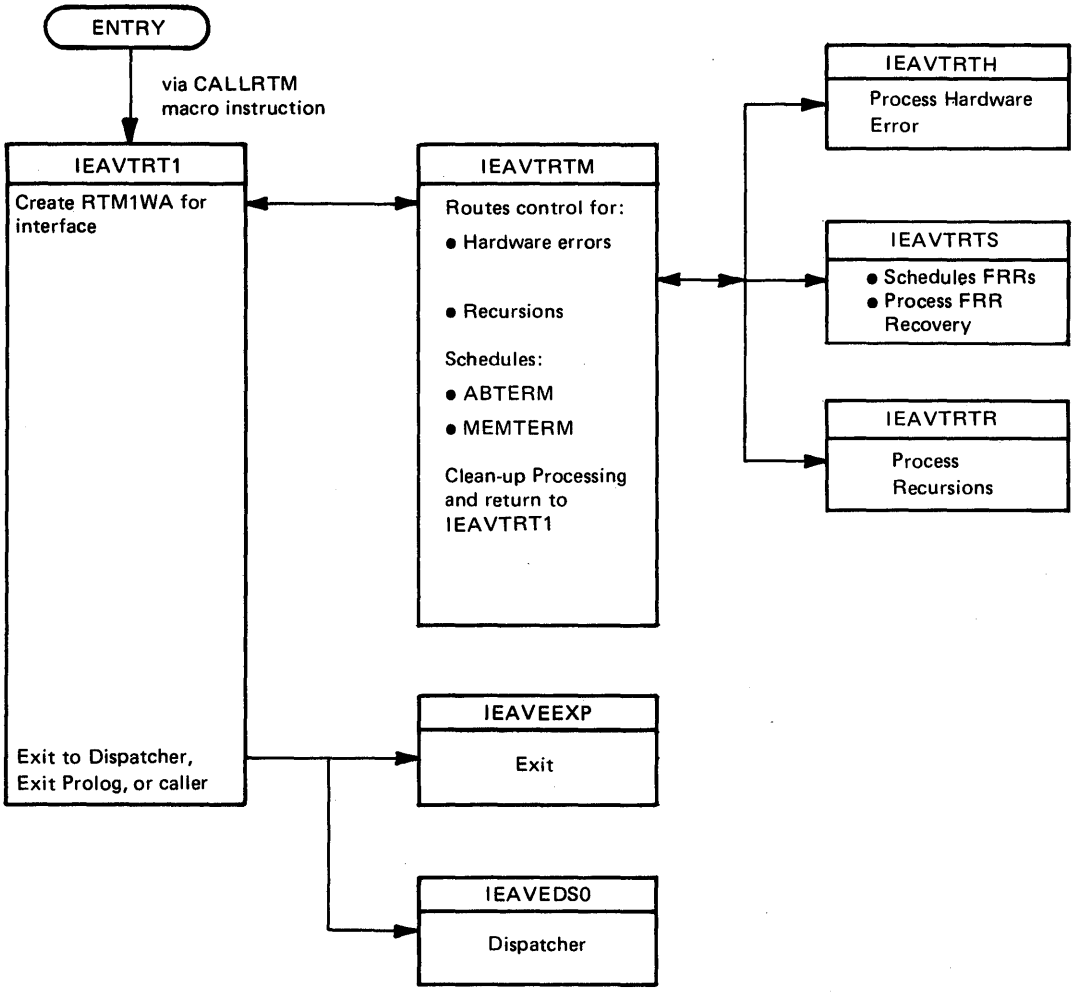


Figure 3-88. RTM1 Module Flow and Basic Functions Performed

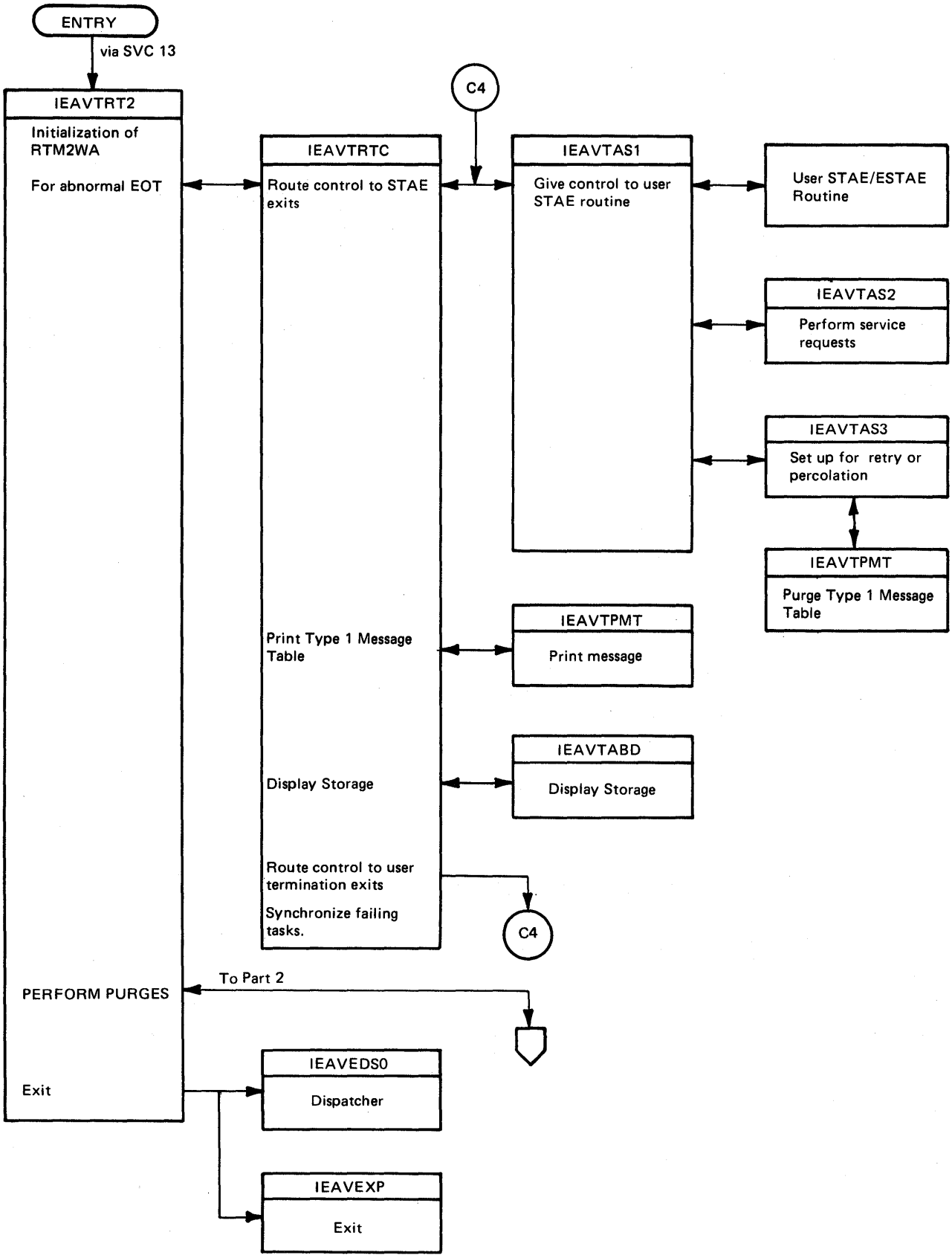


Figure 3-89. RTM2 Module Flow and Basic Functions Performed (Part 1 of 2)

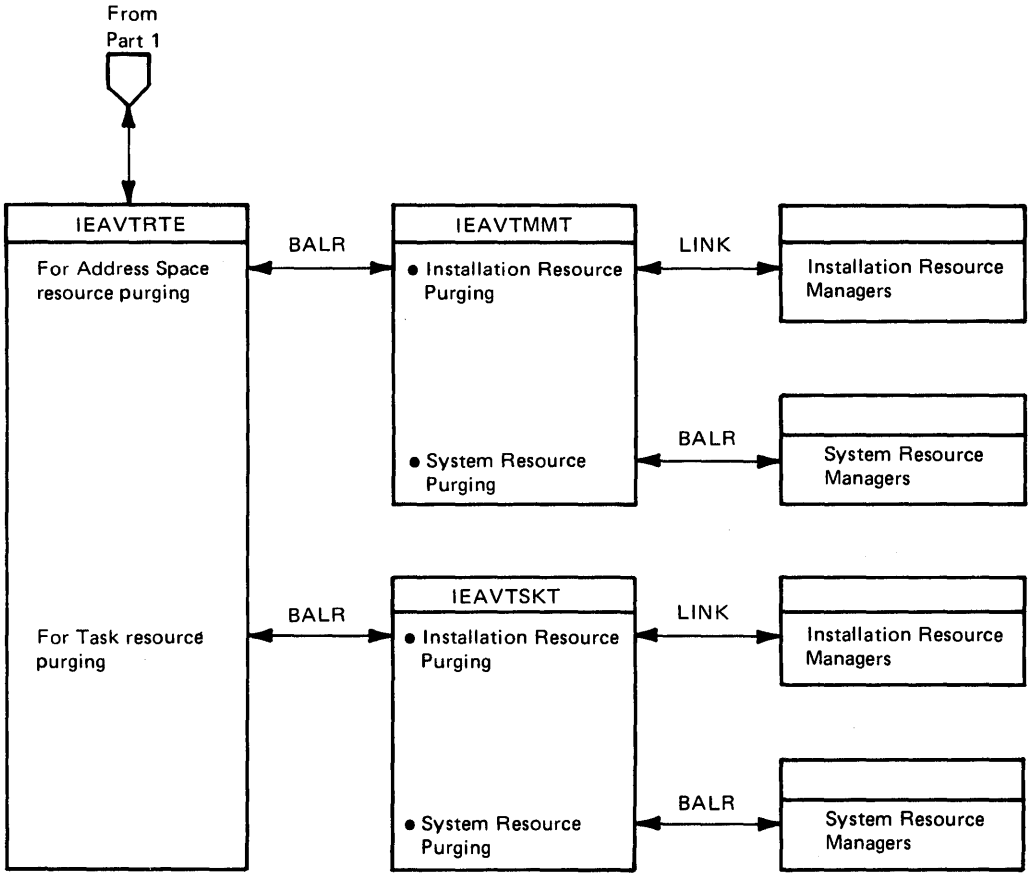


Figure 3-89. RTM2 Module Flow and Basic Functions Performed (Part 2 of 2)

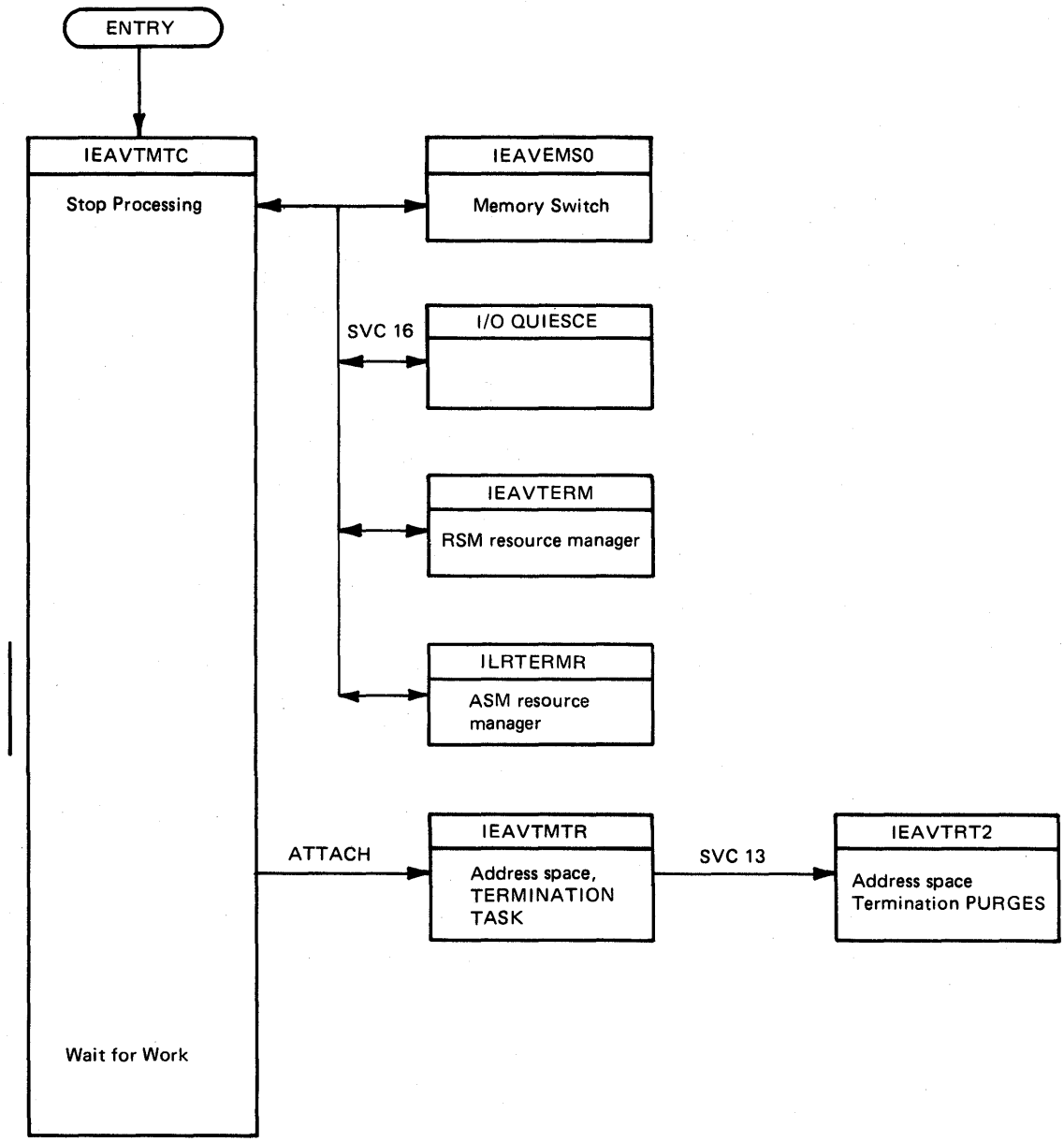


Figure 3-90. Address Space Termination Module Flow

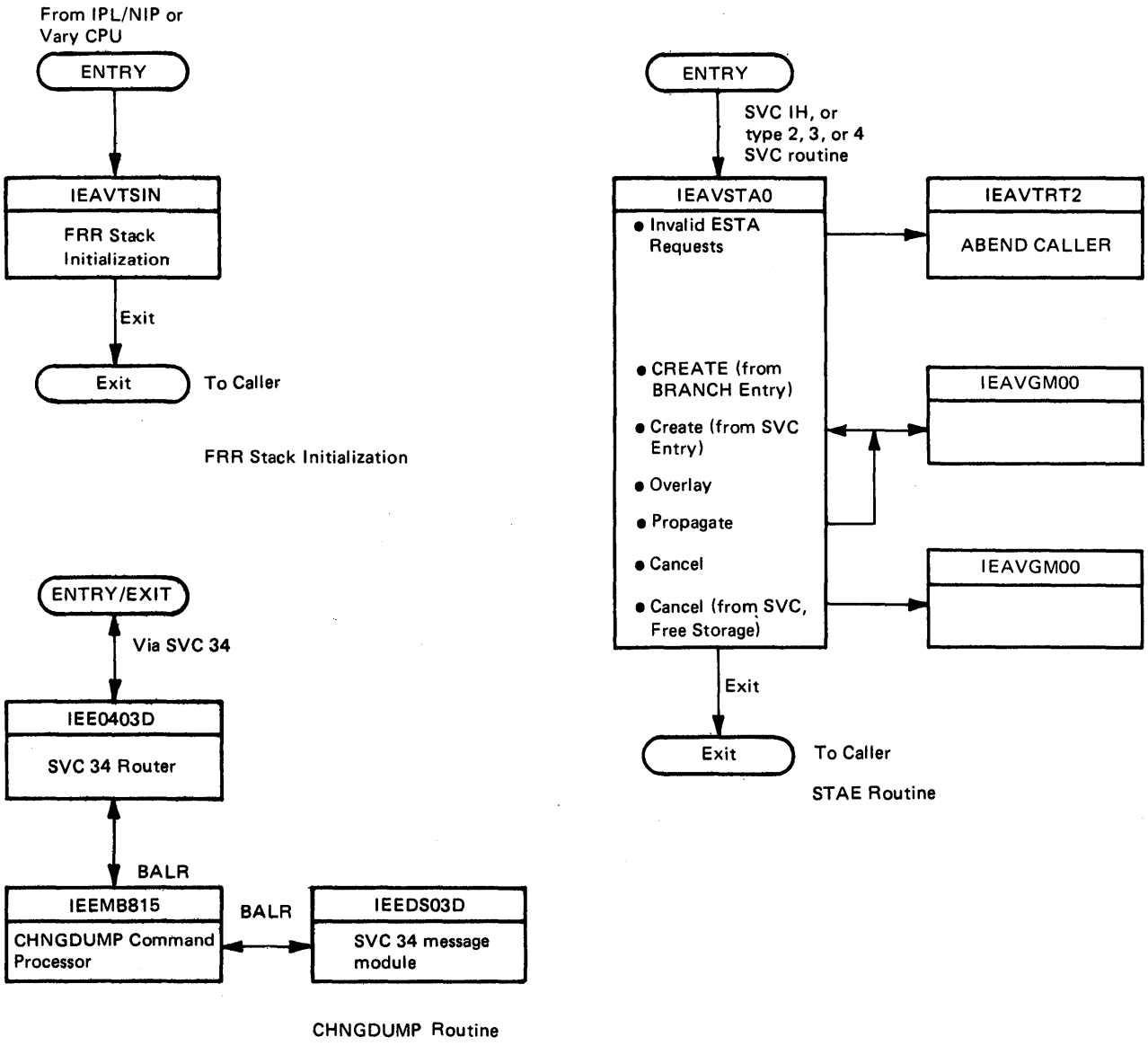


Figure 3-91. R/TM Services Module Flow (Part 1 of 3)

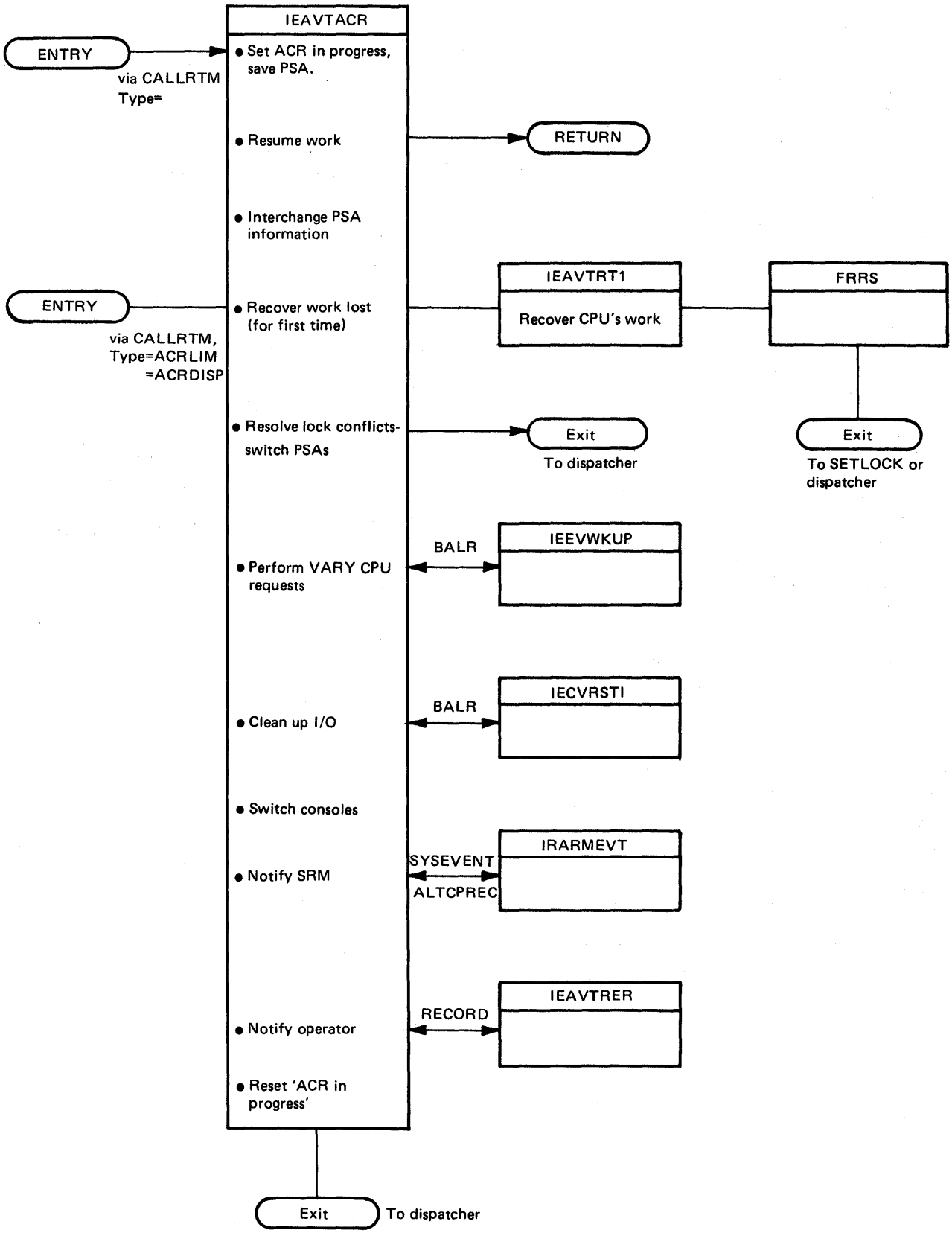


Figure 3-91. R/TM Services Module Flow (Part 2 of 3)

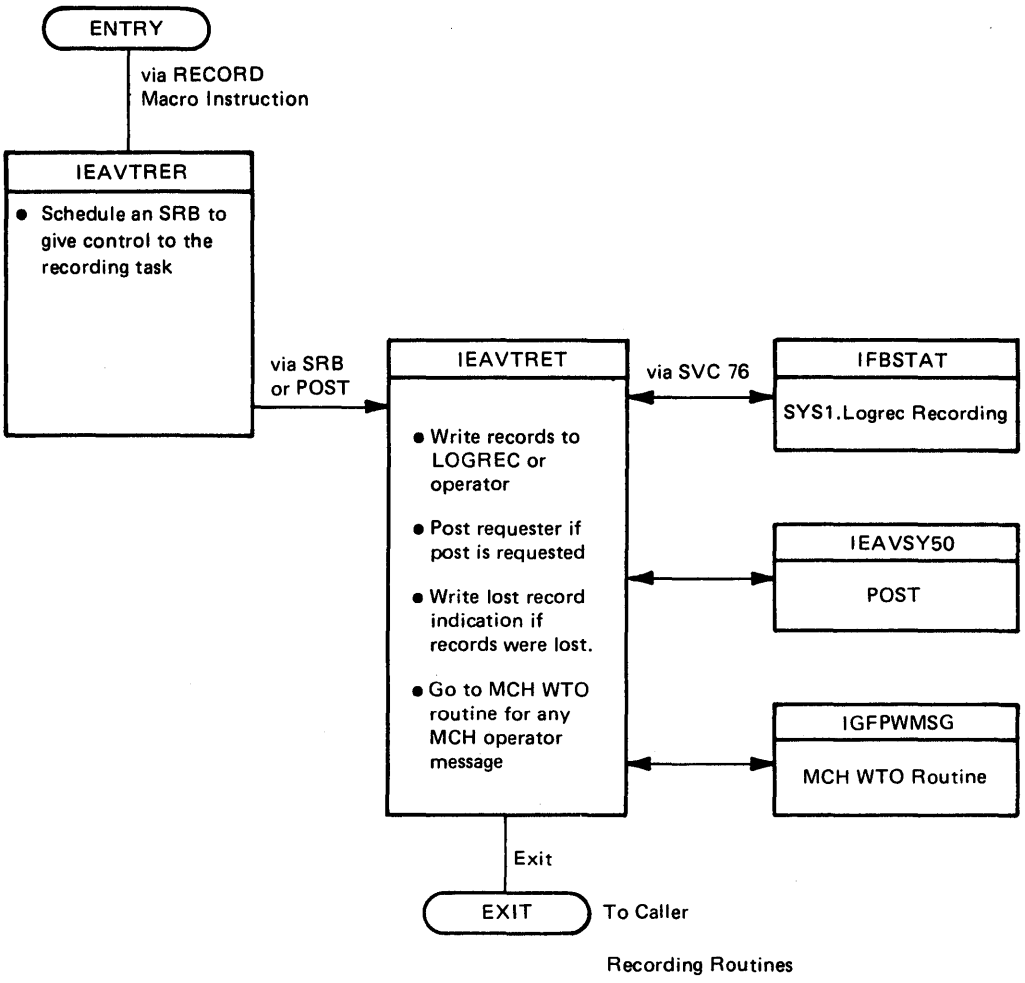


Figure 3-91. R/TM Services Module Flow (Part 3 of 3)

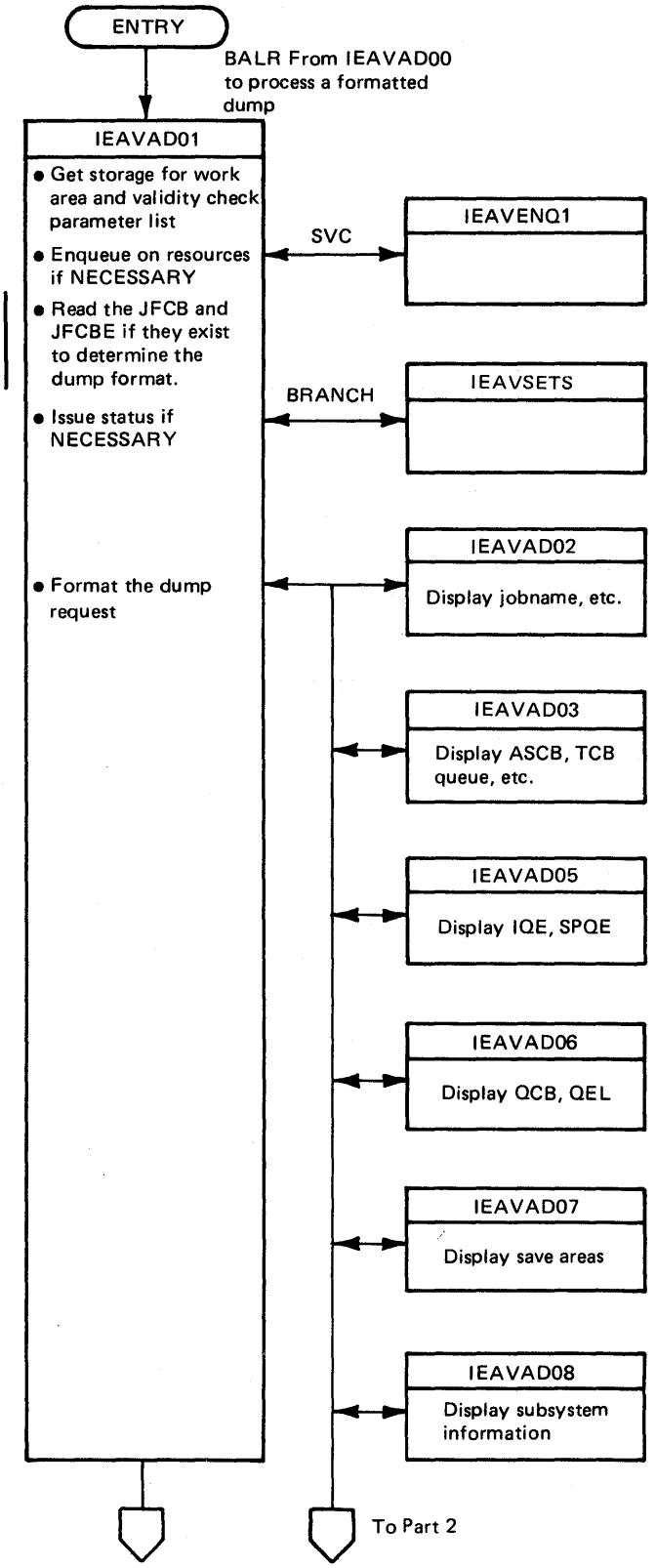


Figure 3-92. R/TM Dumping Services – Formatted Dump Module Flow (Part 1 of 2)

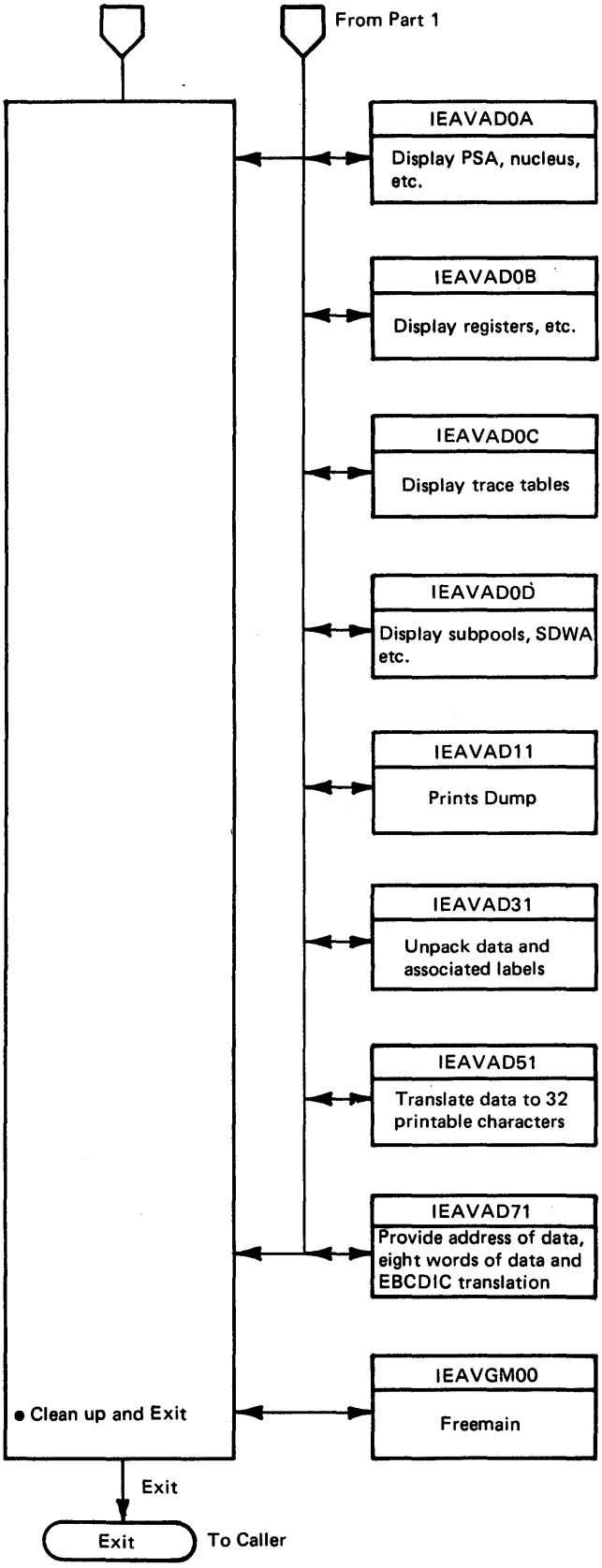


Figure 3-92. R/TM Dumping Services – Formatted Dump Module Flow (Part 2 of 2)

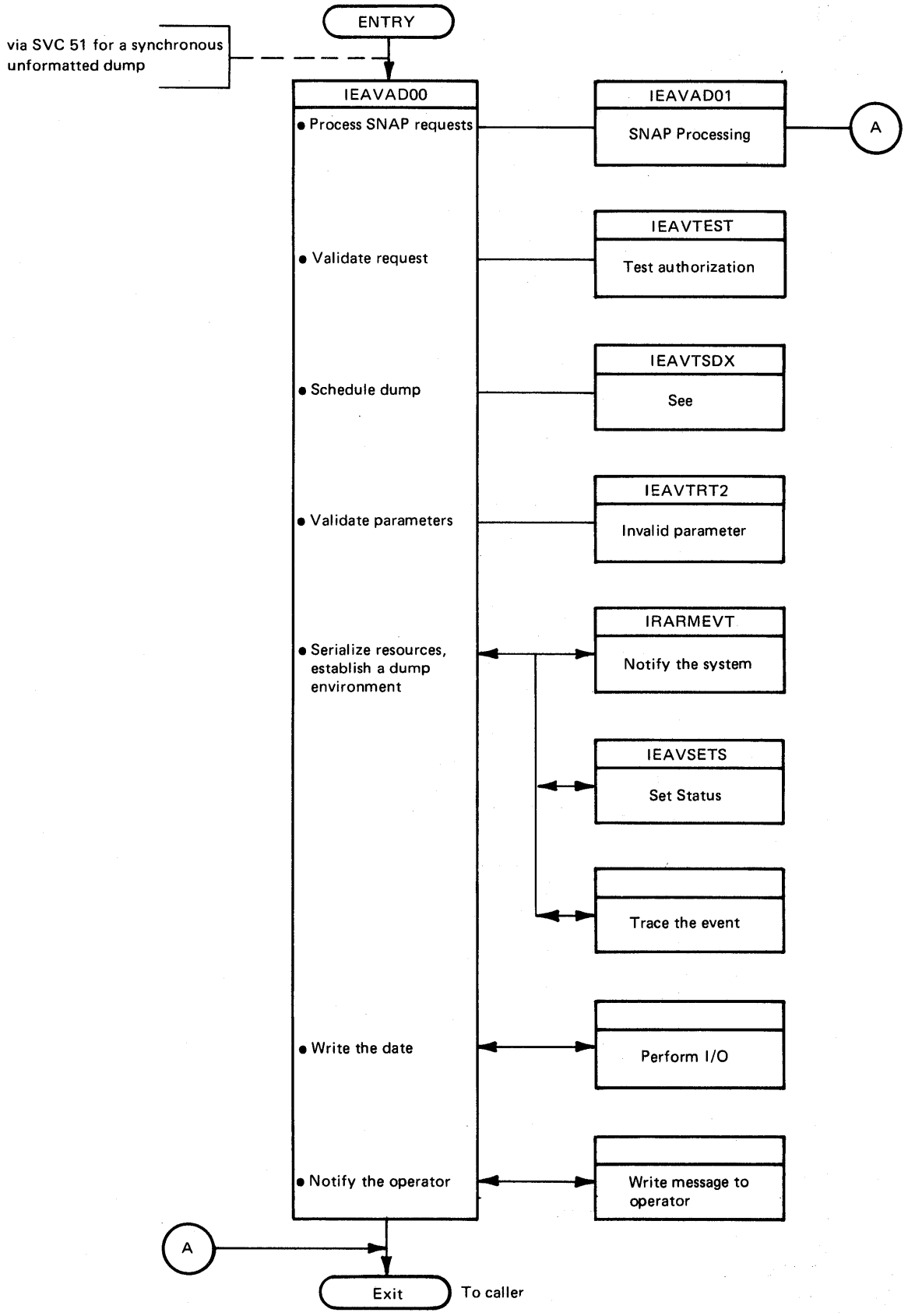


Figure 3-93. R/TM Dumping Services – Synchronous Unformatted Dump Module Flow

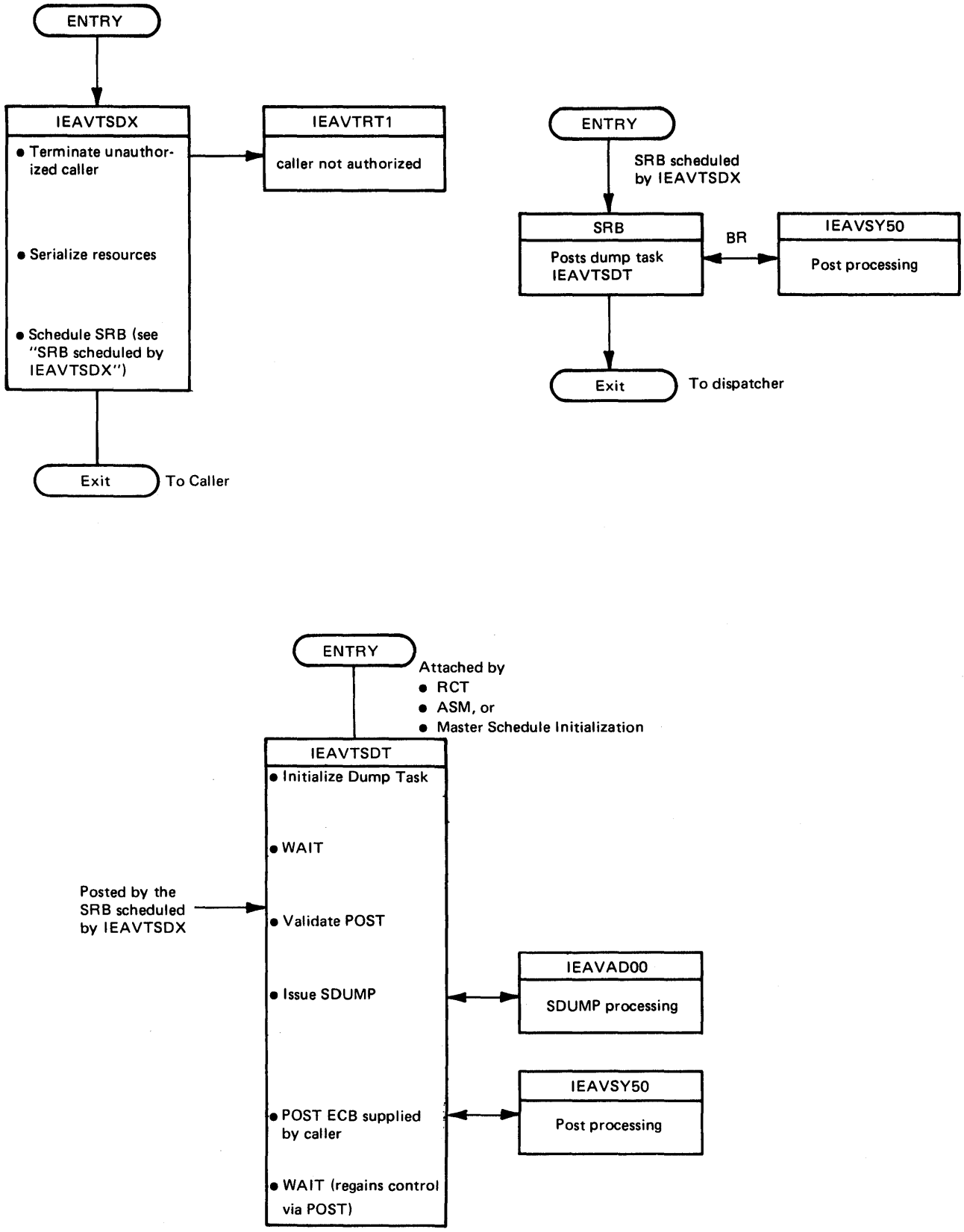
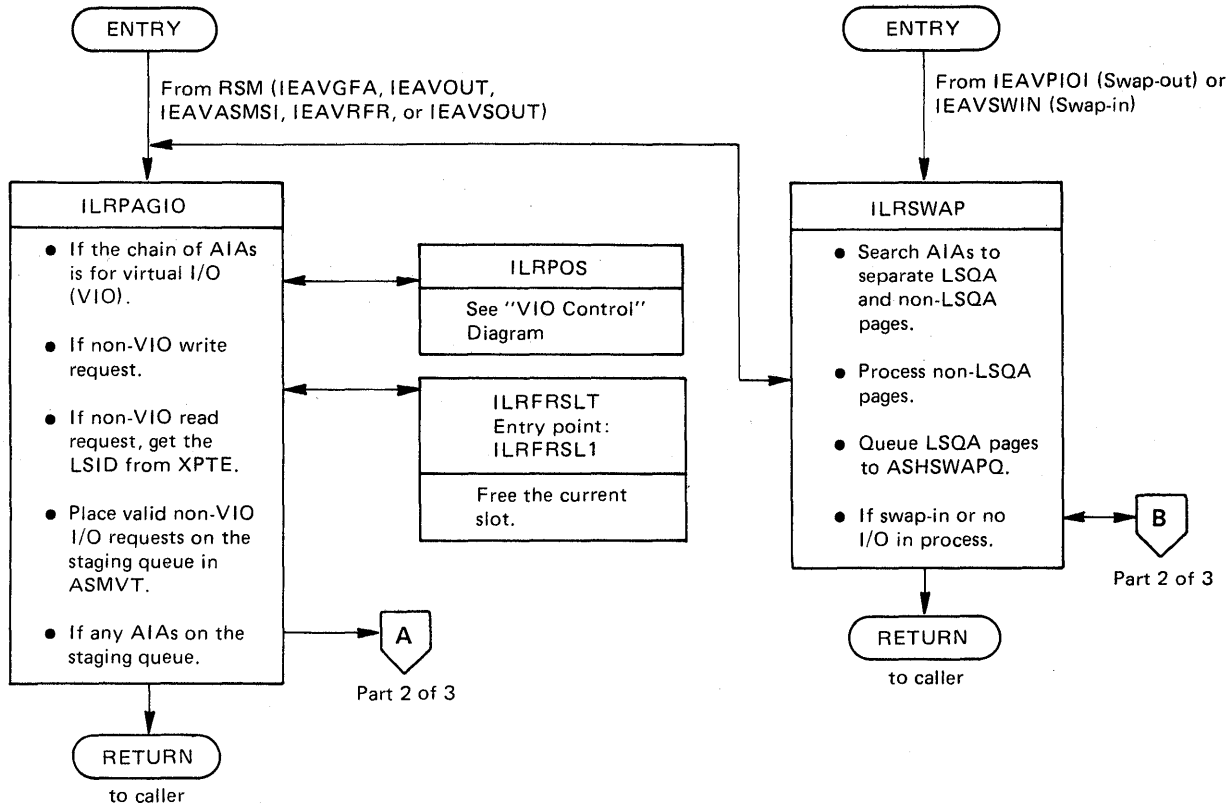
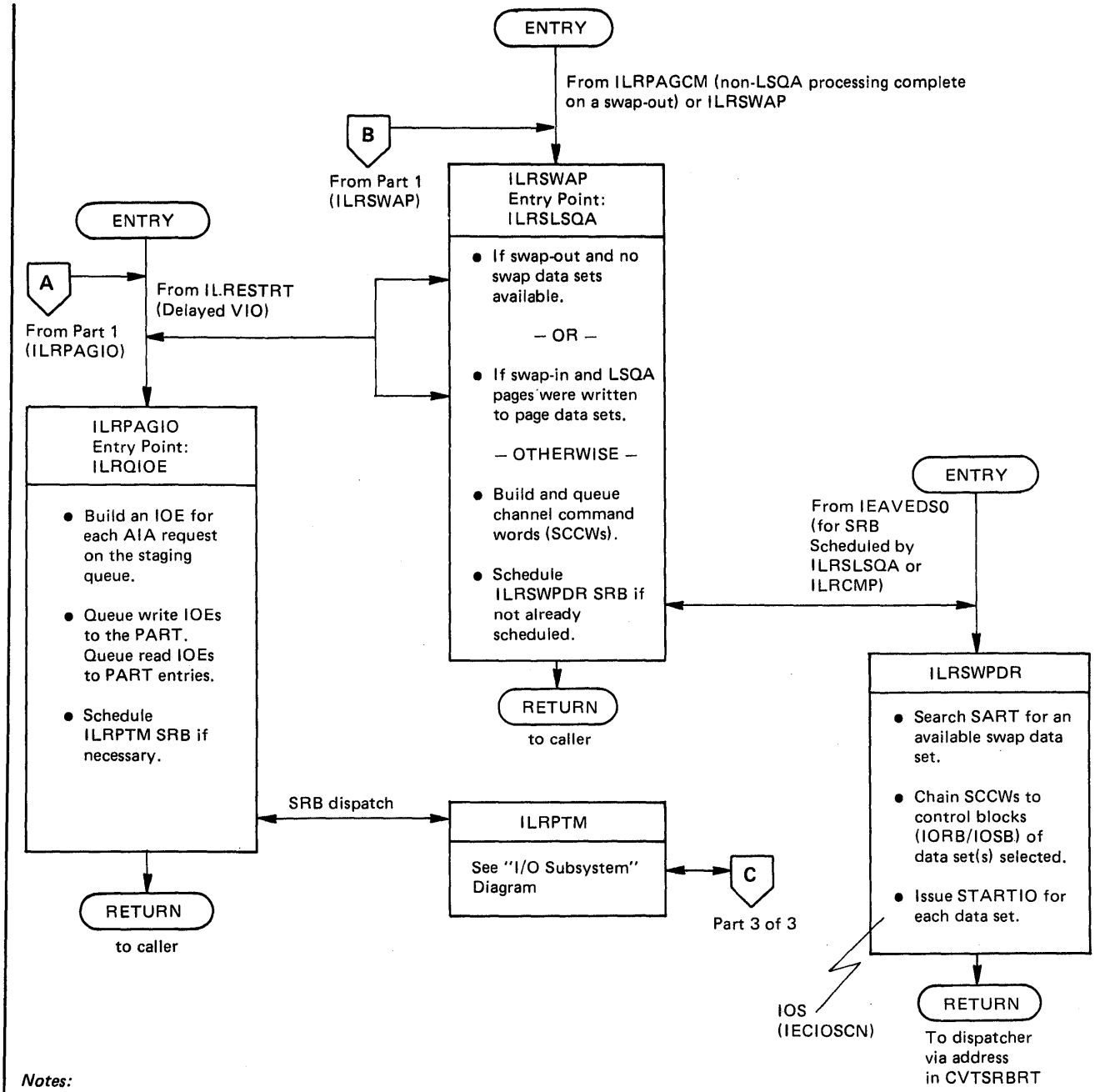


Figure 3-94. R/TM Dumping Services – Scheduled Unformatted Dump Module Flow



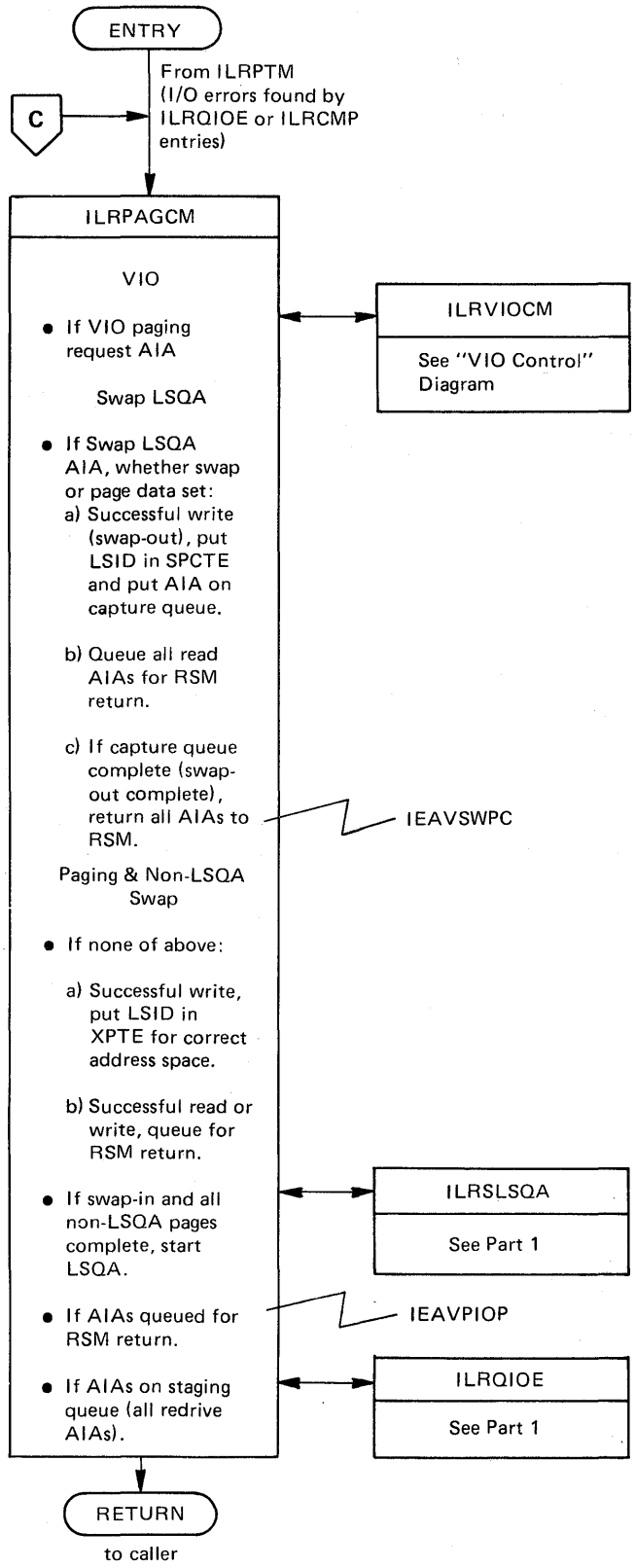
Notes:
1. ILRIOFRR is the recovery routine (FRR) for ILRPAGIO.
2. ILRCSWAP, an entry in ILRSWP01, is the recovery routine for ILRSWAP.

Figure 3-95. I/O Control (Part 1 of 3)



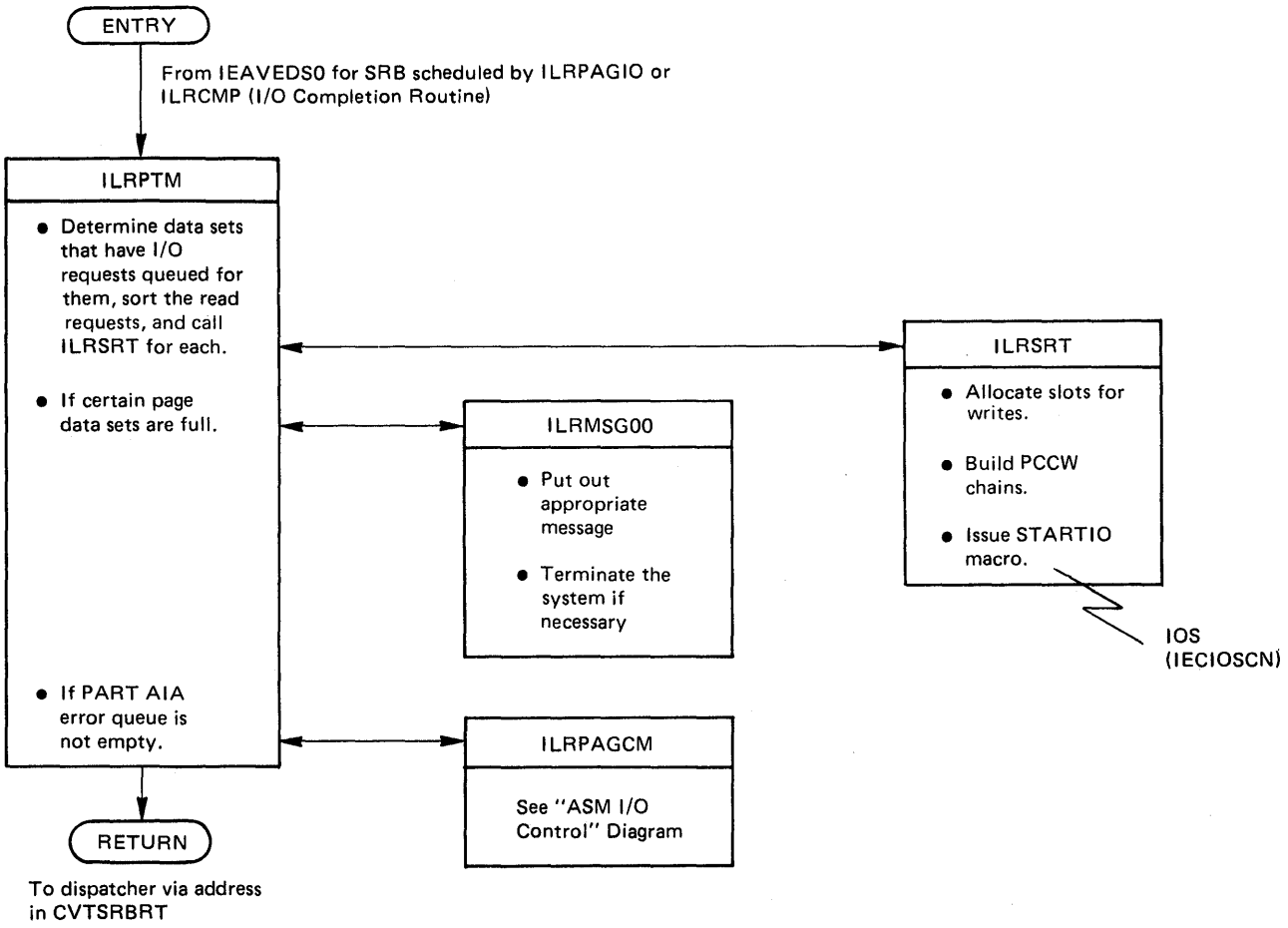
Notes:
1. ILRCQIOE, an entry in ILRIOFRR, is the recovery routine for ILRQIOE.
2. ILRCSLSQ, an entry in ILRSWP01, is the recovery routine for ILRSLSQA.
3. ILRSWP01 is the recovery routine for ILRSWPDR.

Figure 3-95. I/O Control (Part 2 of 3)



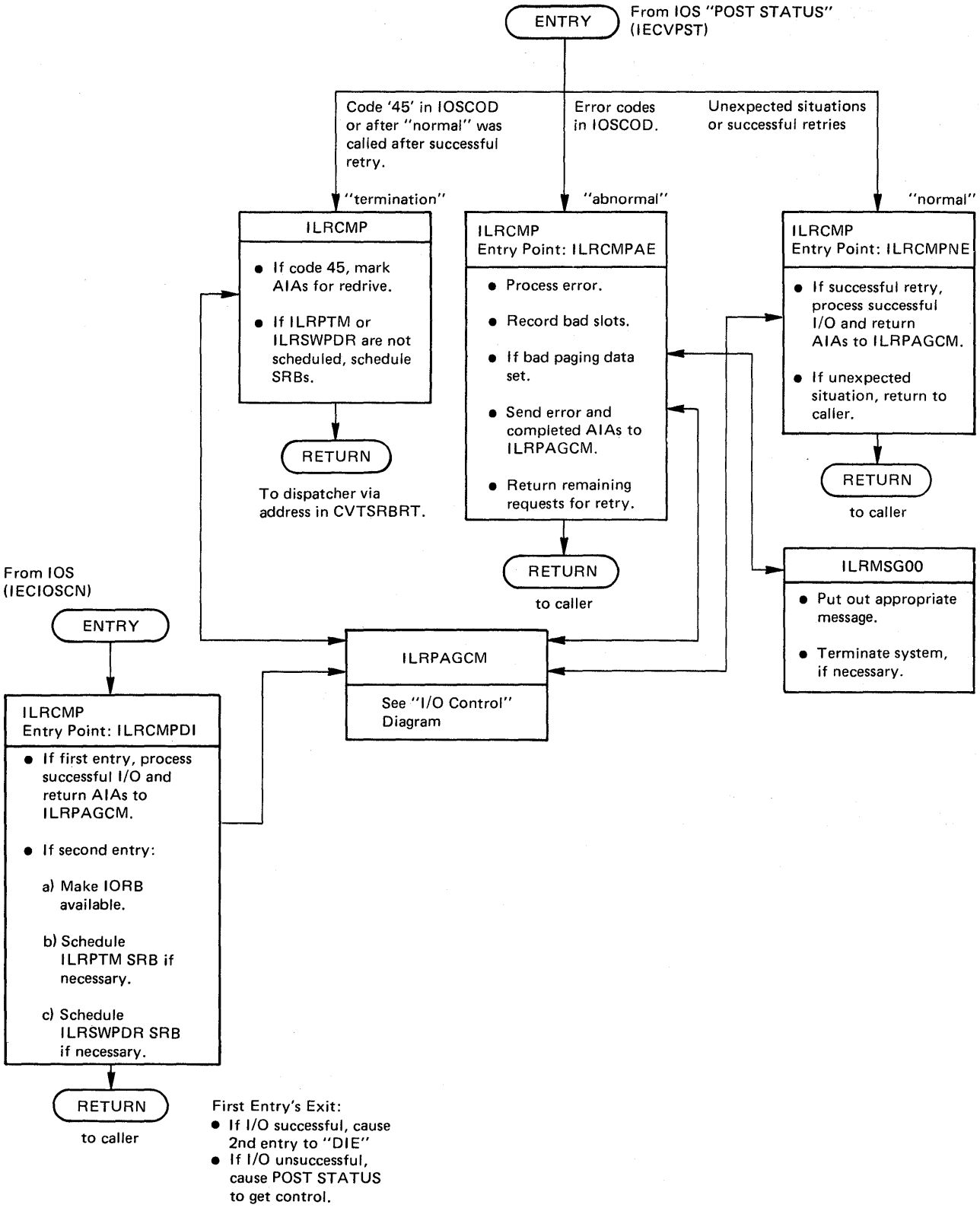
Notes:
1. ILRIOFRR is the recovery routine FRR for ILRPAGCM.

Figure 3-95. I/O Control (Part 3 of 3)



Notes:
1. ILRSRT01 is the recovery routine (FRR) for ILRPTM and ILRSRT.

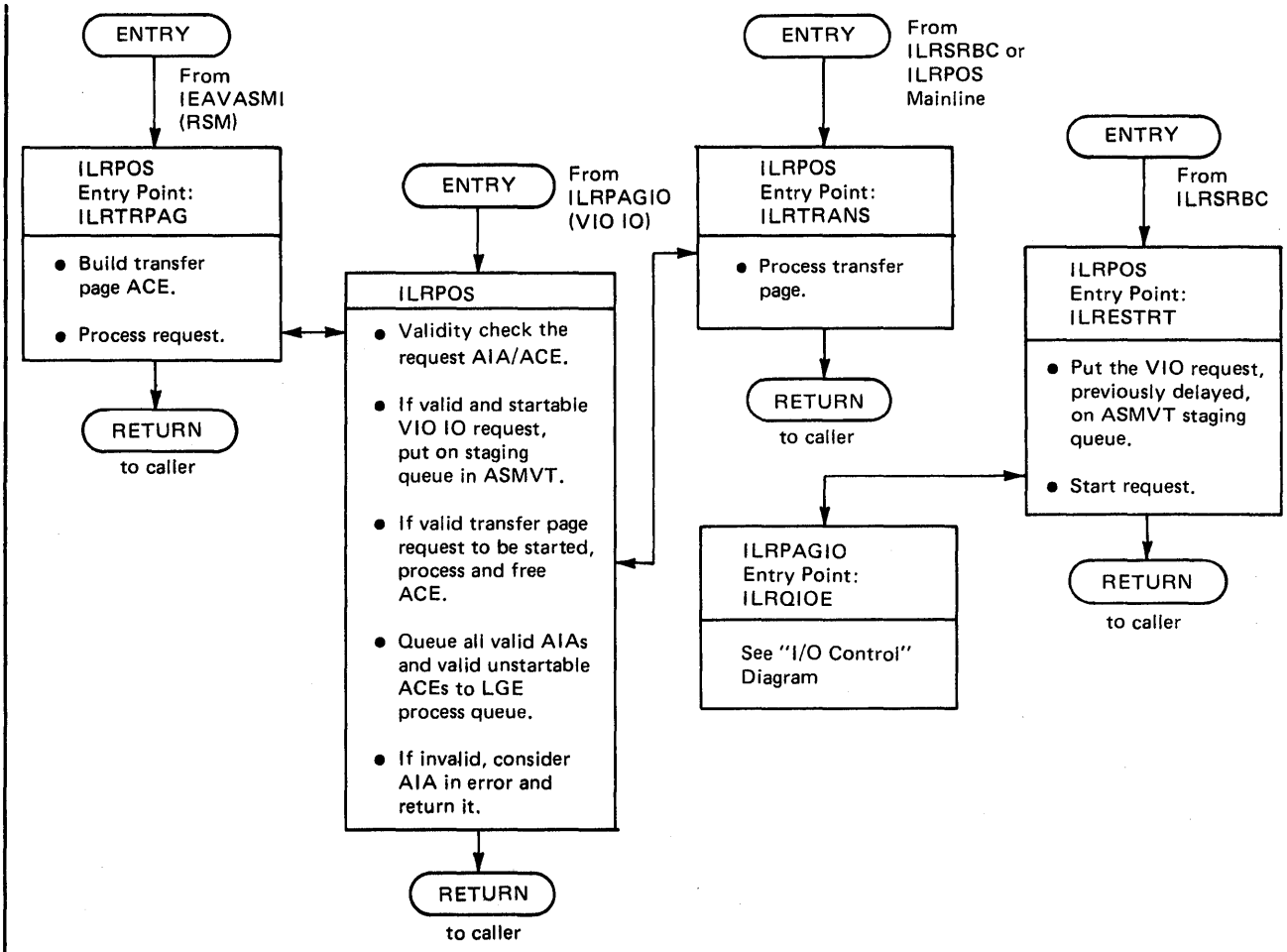
Figure 3-96. I/O Subsystem (Part 1 of 2)



Notes:

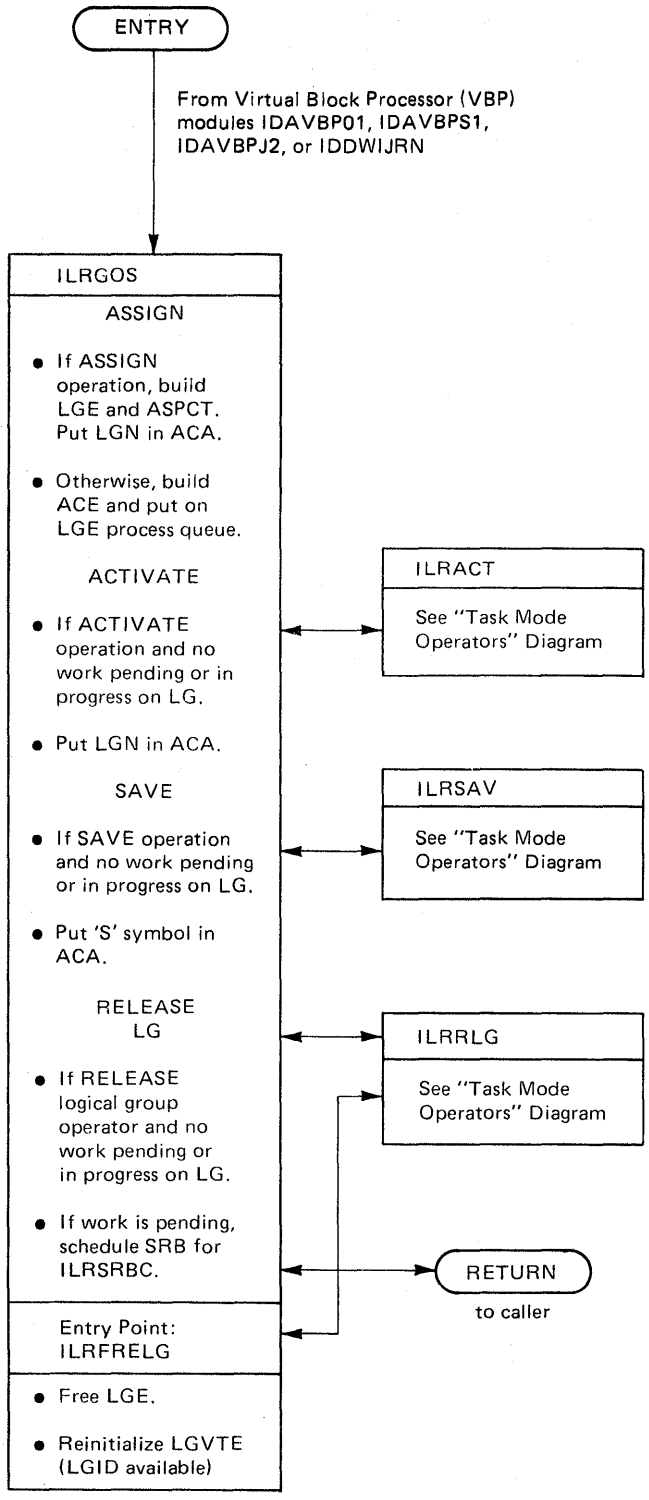
1. ILRCMP01 is the recovery routine (FRR) for ILRCMP.

Figure 3-96. I/O Subsystem (Part 2 of 2)



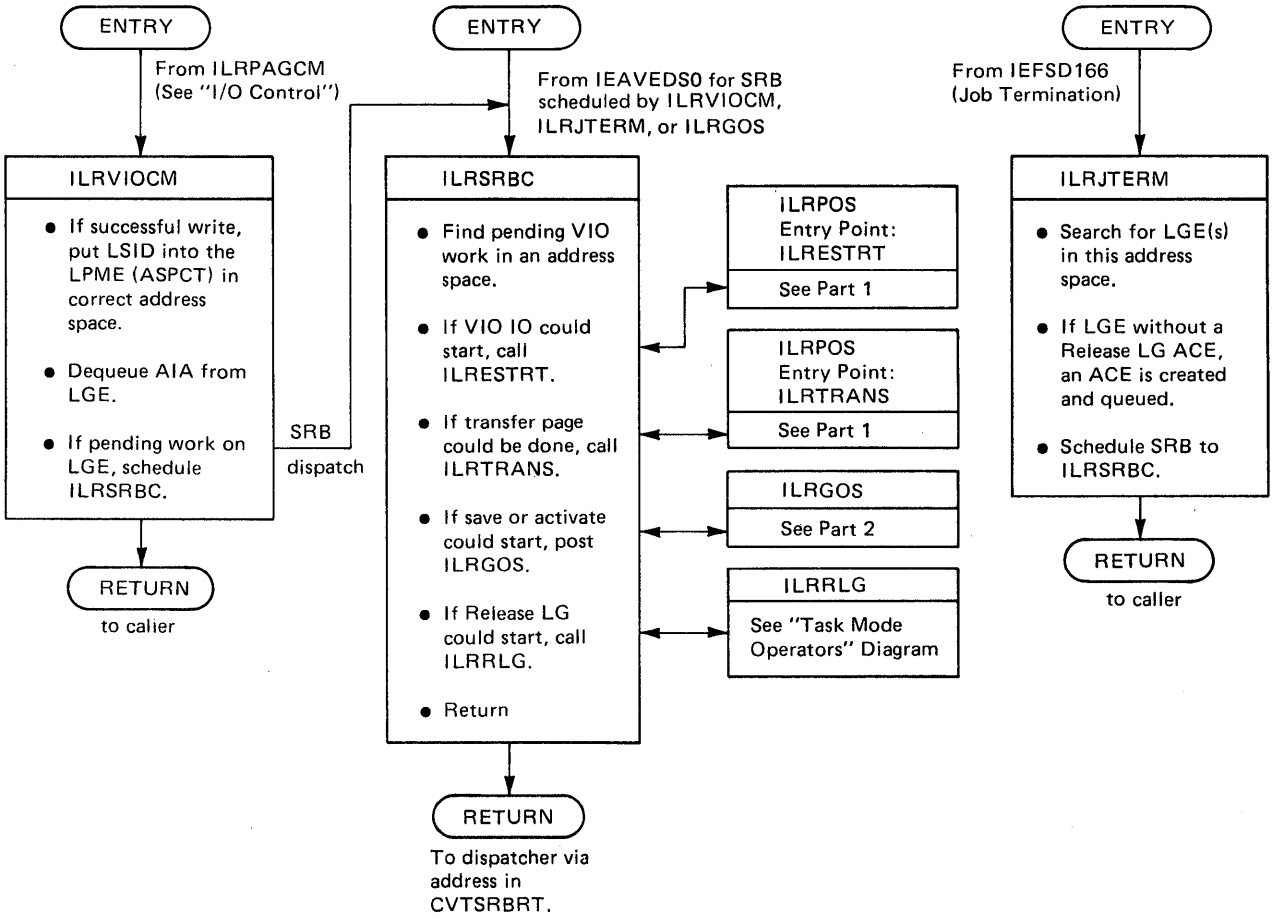
Notes:
1. ILRIOFRR is the recovery routine (FRR) for ILRPOS.

Figure 3-97. VIO Control (Part 1 of 3)



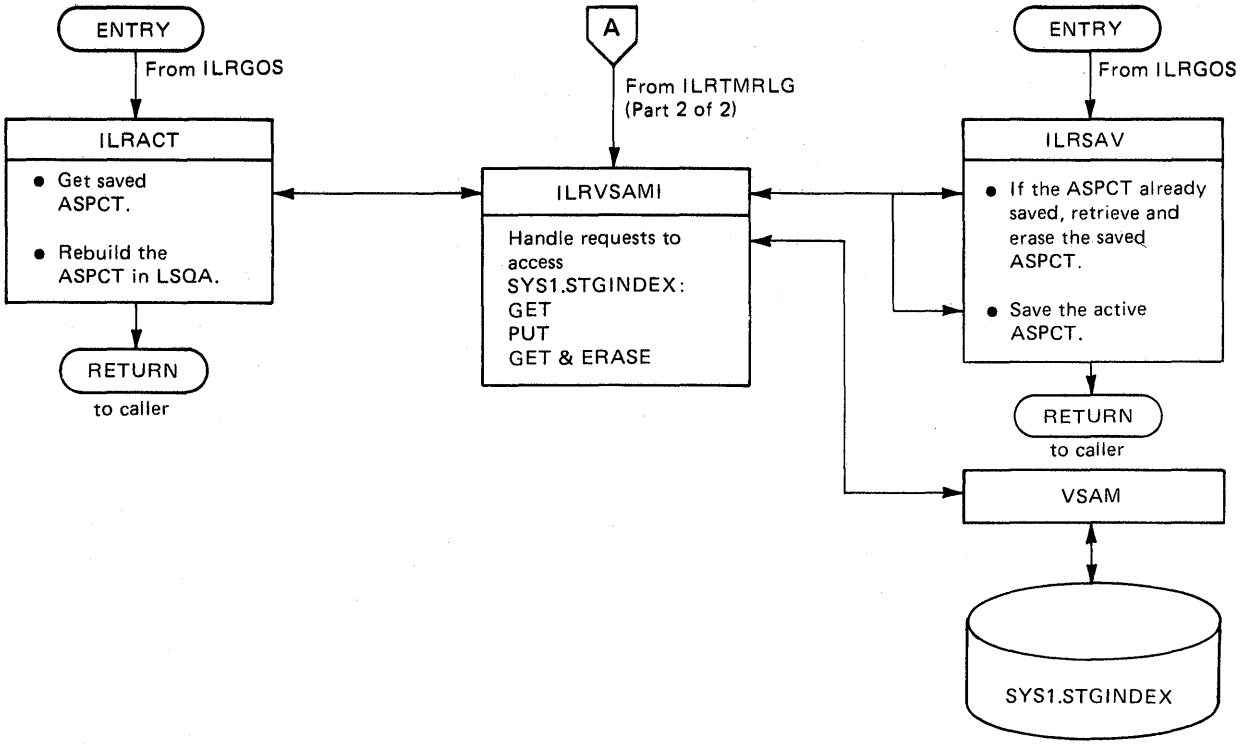
Notes:
1. ILRGOS01 is the recovery routine (FRR) for ILRGOS.

Figure 3-97. VIO Control (Part 2 of 3)



Notes:
1. ILRIOFRR is the recovery routine (FRR) for ILRVIOCM.
2. ILRSRB01 is the recovery routine (FRR) for ILRSRBC.
3. ILRJTM01 is the recovery routine (FRR) for ILRJTERM.

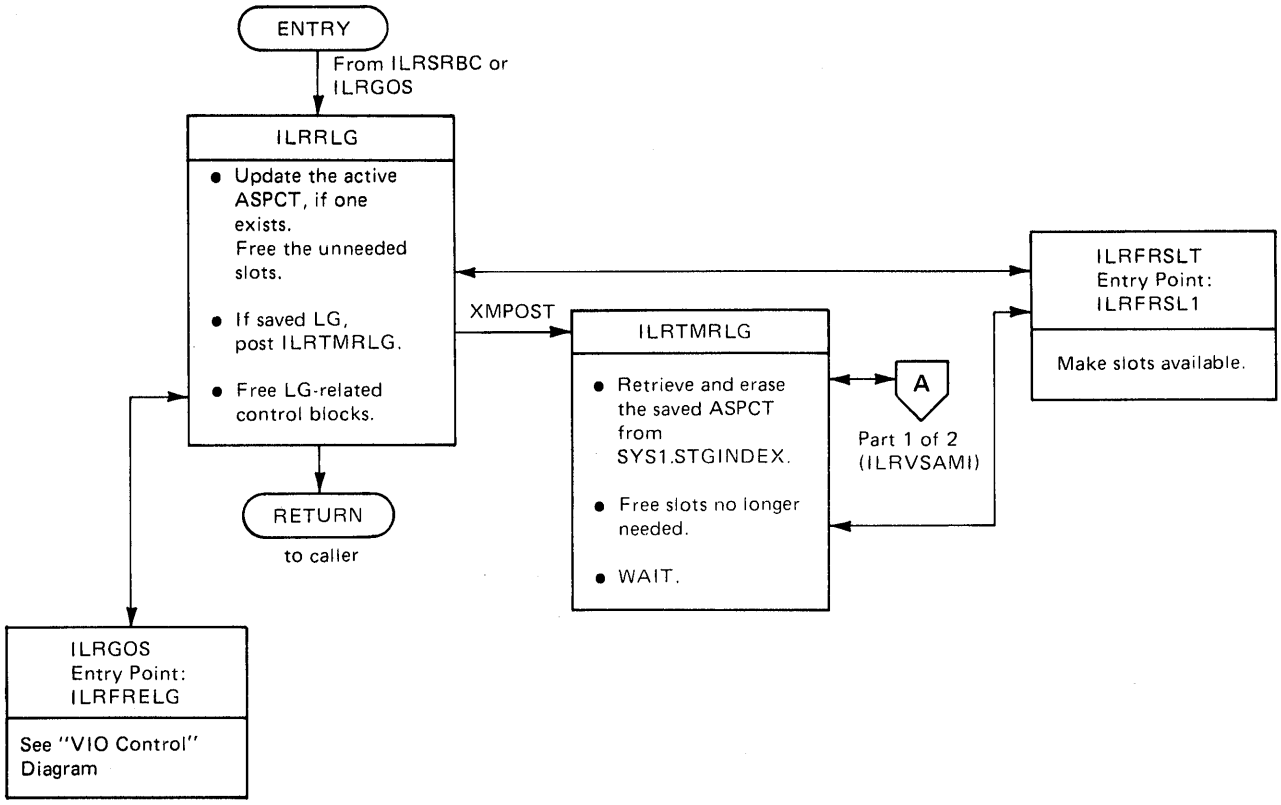
Figure 3-97. VIO Control (Part 3 of 3)



Notes:

1. ILRGOS01 is the recovery routine (ESTAE) for ILRACT, ILRSAV, and their paths through ILRVSAMI.

Figure 3-98. VIO Group Operators (Part 1 of 2)



Notes:

1. ILRGOS01 is the recovery routine (FRR) for ILRRLG.
2. ILRTMI01 is the recovery routine (ESTAE) for ILRTMRLG and its path through ILRVSAMI.

Figure 3-98. VIO Group Operators (Part 2 of 2)

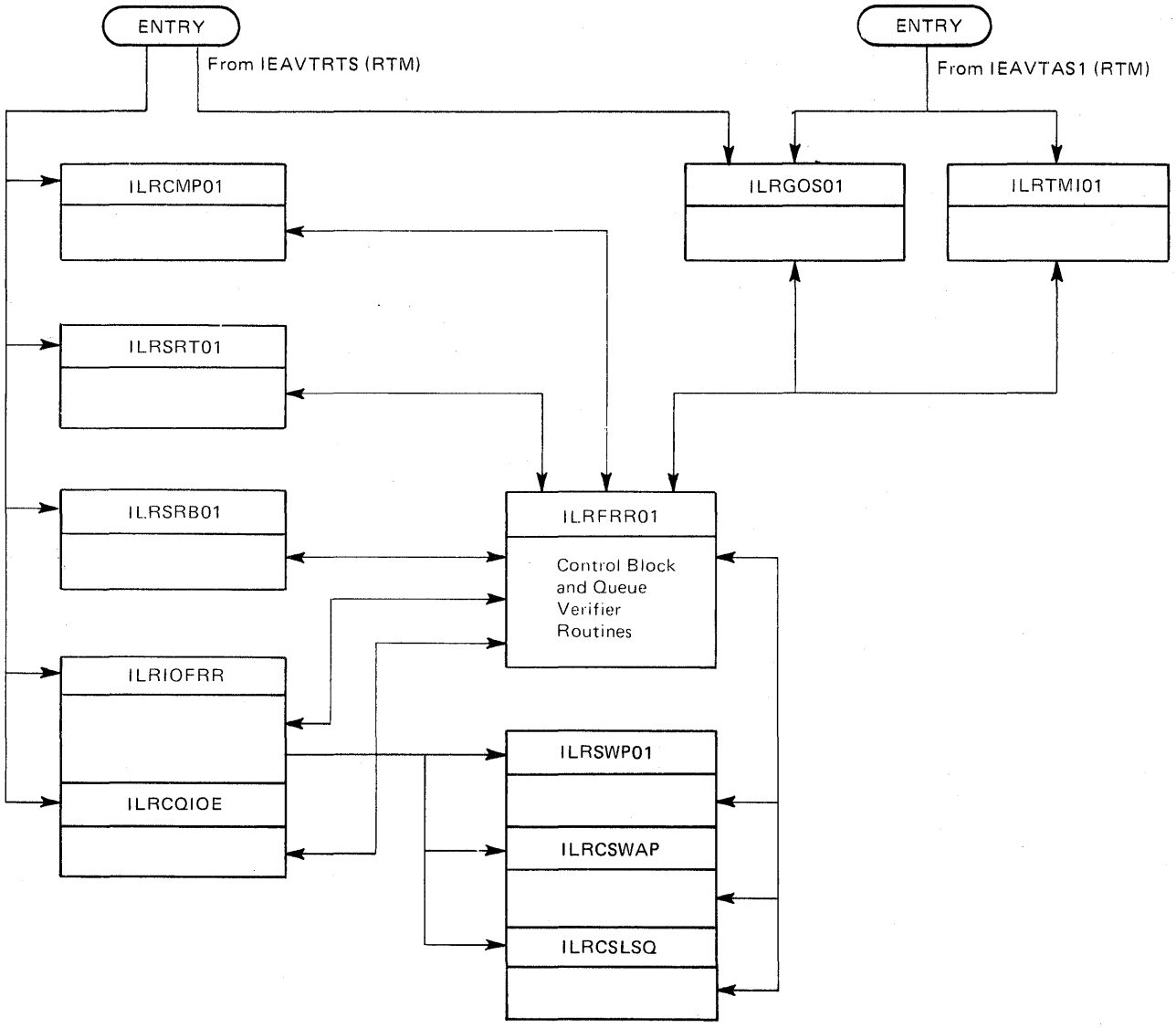
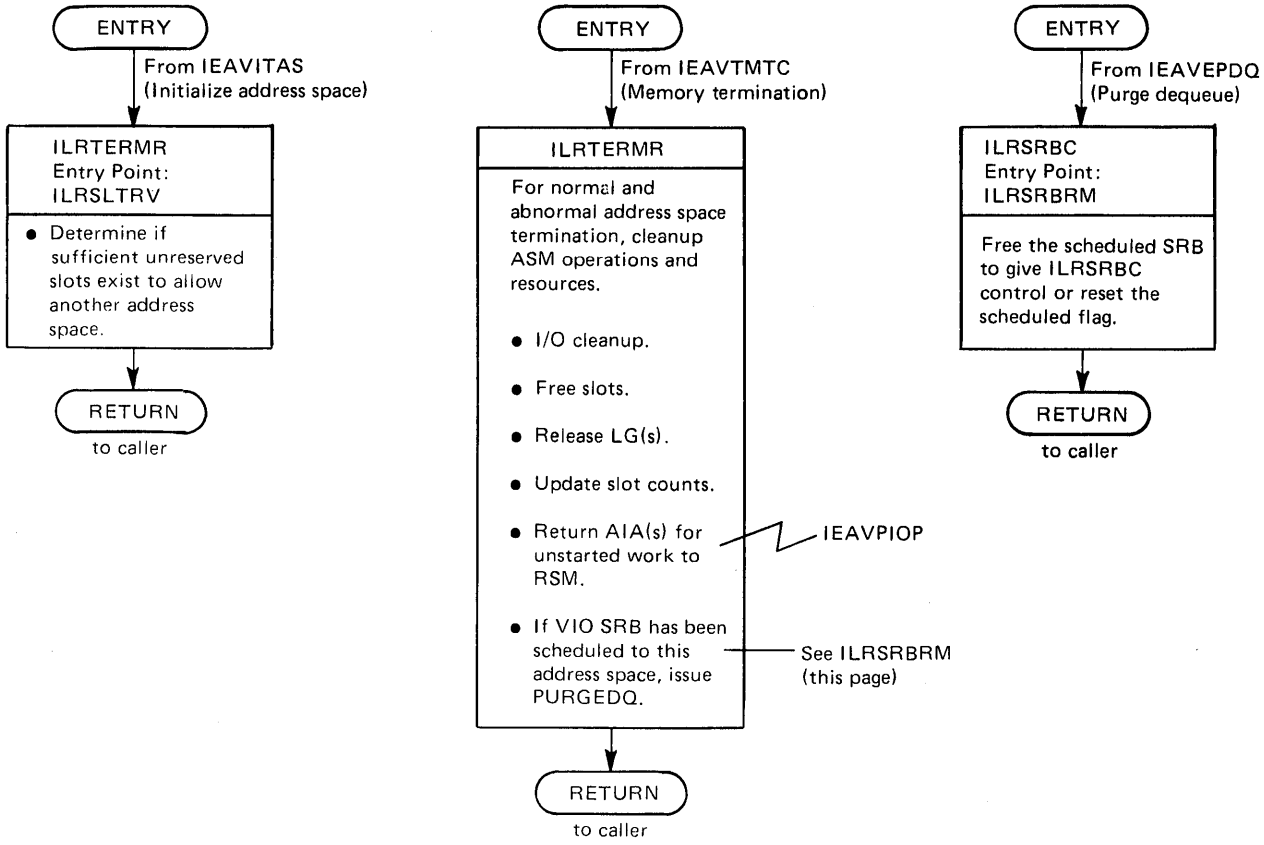


Figure 3-99. Recovery



Notes:

1. TERMRFRR is the recovery routine (FRR) for ILRTERMR.

Figure 3-100. Service Routines (Part 1 of 2)

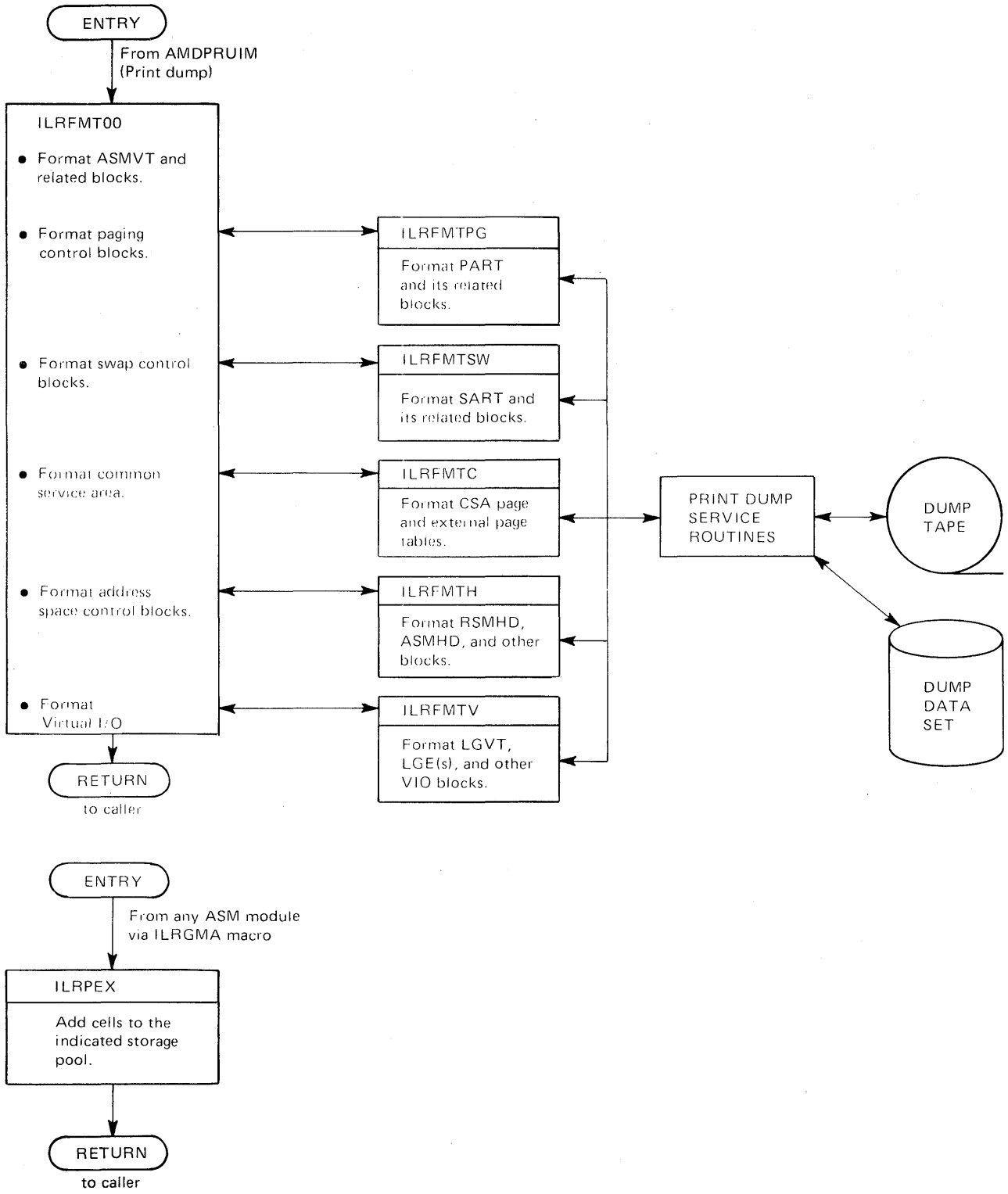


Figure 3-100. Service Routines (Part 2 of 2)

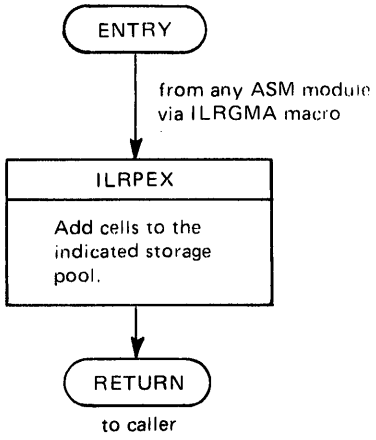


Figure 3-100. Service Routines (Part 3 of 3)

This blank leaf represents pages 6-214 - 6-255 which were deleted by Supervisor Performance #2.

Module Descriptions

If there are two names given in a title, the first in the CSECT name and the second (in parentheses) is the assembler module. If there is only one name in the title, it is the same for both the CSECT and assembler module.

IEATLEXT — SMF Time Limit Extension

Operation: This module is entered if a job, job step, or wait-time limit has expired. It interfaces with the SMF user's exit routine, IEFUTL. If the exit grants an extension, IEATLEXT extends the expired limit by the amount specified by the exit. If the exit does not grant an extension, IEATLEXT will abend the expired task with a code 322 for a job or job step limit expiration, or 522 for a wait-time limit expiration.

Entry from: Dispatcher (IEAVEDS0).

Exit to: Dispatcher (IEAVEDS0).

IEAV1052 — 1052 Console Device Support Processor

Operation: Performs the following functions for a 1052 printer-keyboard, 3210 console printer-keyboard, 3215 console printer-keyboard, or 3213 console printer used as a console

- Opens the device as a console.
- Initiates a read operation to read a command from the console.
- After a read operation completes successfully, passes the command to command processing (SVC 34).
- Initiates a write operation to write messages to the console.
- After a write operation completes successfully, updates the console queue.
- When the device is to be removed from console status, closes the console.

Entry from: IEAVVCTR.

Exit:

- To IEAVSWCH if an I/O error occurs or the console cannot be opened.
- Otherwise, to issuer of SVC 72.

IEAV1443 — 1443 Console Device Support Processor

Operation: Performs the following functions for a 1443 printer, 1403 printer, or 3211 printer used either as the output portion of a composite console or as an output-only console

- Opens the printer as an output console.
- Uses BSAM to write WTO and WTOR messages to the console.
- After a write operation completes successfully, updates the console.
- When the printer is to be removed from console status, closes the console.

Entry from: IEAVVCTR.

Exit:

- To IEAVSWCH if an I/O error occurs or the console cannot be opened.
- Otherwise, to the issuer of SVC 72.

IEAV2540 — 2540 Console Device Support Processor

Operation: Performs the following functions for a 2540 card reader punch, 2501 card reader, 2520 card reader punch, 3505 card reader, or 3525 card punch used as the input portion of a composite console

- Opens the device as an input console.
- Uses BSAM to read a command from the console.
- After a read operation completes successfully, passes the command to command processing (SVC 34).
- When the device is to be removed from console status, closes the console.

Entry from: IEAVVCTR.

Exit:

- To IEAVSWCH if an I/O error occurs or the console cannot be opened.
- Otherwise, to the issuer of SVC 72.

IEAVAD0A — Nucleus, SQA, LSQA

Operation: This module displays the control program nucleus, allocated SQA, and LSQA.

Entry from: IEAVAD01 via BALR.

Exit to: IEAVAD01.

IEAVAD0B — Regs, SNAP, LPA, JPA, Active SVC Modules

Operation: This module conditionally displays the registers at entry to SNAP or ABEND, the storage represented by the list supplied by the caller, those LPA/JPA modules that are on the active RB queue or load list of the task being dumped, and the active SVC modules for the task.

Entry from: IEAVAD01 via BALR.

Exit normal: To caller.

Exit error: To caller.

IEAVAD0C — Supervisor Trace Formatting

Operation: This module displays the supervisor trace table.

Entry from: IEAVAD01 via BALR

Exit normal and error: To caller.

IEAVAD0D — Display Allocated Space Within User Subpools and SWA

Operation: This module displays allocated space in user subpools subpool 252, and SWA. This module is also called to display subpools 229 and 230 when LSQA is requested.

Entry from: IEAVAD01 via BALR.

Exit-normal: To caller.

Exit-error: To caller.

IEAVAD00 — SVC 51

Operation: This module identifies which function is requested (SNAP or SVC DUMP) and routes control to the corresponding routine. The SVC DUMP ROUTINE is contained in the module.

Entry from: SVC IH.

Exit-normal: To Exit Routine.

Exit-error: Abend caller.

IEAVAD01 — SNAP Mainline

Operation: This module gets storage for, and initializes, buffers, and validity checks SNAP parameter list. It also routes control to the appropriate functional display routine and cleans up after all indicated areas have been displayed.

Entry from: IEAVAD00 via BALR.

Exit to: IEAVAD00.

IEAVAD02 — Header, PSW Formatting

Operation:

This module formats and prints the dump ID, jobname, stepname, time, completion code, PSW, ILC, and interruption code.

Entry from: IEAVAD01 via BALR.

Exit normal and error: To caller.

IEAVAD03 — Control Blocks I

Operation: This module formats and prints the ASCB, TCB, RBS, LLES, CDES, extent lists, DEBS, and the TIOT for the task being dumped.

Entry from: IEAVAD01 via BALR

Exit normal: To caller.

Exit error: To caller.

IEAVAD05 — Control Blocks II

Operation: This module formats and displays the VMM blocks (SPQE, DQE, and FQE), the IQE, PQE, and FBQE queues.

Entry from: IEAVAD01 via BALR.

Exit normal and error: To caller.

IEAVAD06 — QCB and QEL Dump

Operation: This module formats and displays the queue elements representing resources belonging to the task being dumped.

Entry from: IEAVAD01 via BALR.

Exit normal and error: To caller.

IEAVAD07 — Display Save Areas

Operation: This module provides the display of user save areas.

Entry from: IEAVAD01 via BALR.

Exit normal and error: To caller.

IEAVAD08 — SNAP, TCAM, GTF, VSAM, and Shared Resources Interface Module

Operation: This module is responsible for the interfaces to the TCAM, GTF, VSAM, and shared resources formating routines.

Entry from: IEAVAD01 via BALR.

Exit to: IEAVAD01. IDA0195A via BALR for VSAM. IGA0E05A via BALR for TCAM. IGC0F05A via BALR for GTF. IGC0905A via BALR for Shared Resources. ISTRAFD1 via BALR for VTAM.

IEAVAD11 — SNAP Output Routines

Operation: This module writes the lines of the dump on the output device.

Entry to: IEAVAD21: IEAVAD21 forces output of any records remaining in buffer after dump is complete.
Entry to: IEAVAD81: IEAVAD81 is the interface between the subsystem formatting routines and the output module IEAVAD11.

Entry from: IEAVAD01 via BALR.

Exit to: Caller.

IEAVAD31 — SNAP Format and Format 01 Routines

Operation: This module unpacks data into the output line while it associates a label with each piece of data.
Entry to: IEAVAD31: This entry converts the data to decimal notation and unpacks it into the output line associating labels with the fields. It also allows an indentation factor to location in an output line where the data is to be placed.
Entry to: IEAVAD41: This entry performs the same functions as IEAVAD41 except that no identation factor is anticipated in the layout line.

Entry from: Caller via BALR.
Exit-normal for entries IEAVAD31 and IEAVAD41: Back to caller.
Exit-error for entries IEAVAD31 and IEAVAD41: A 0C4 ABEND will occur if an unexpected UPR occurs.

IEAVAD51 — SNAP Format 20 and Format 22 Routines

Operation: Format 20 translates the line into printable characters and unpacks 3 bytes of data into the beginning of the line. Format 22 unpacks up to four bytes of data into any positions of the line.

Entry from: IEAVAD51 — Format 20 Routine.

Entry from: IEAVAD61 — Format 22 routine.

Entry from: IEAVAD71 via BALR.

Exit-normal: Back to IEAVAD71.

Exit-error: Back to IEAVAD71 with register 15 set to 8.

IEAVAD71 — SNAP Format Router

Operation: This routine controls the routing of other format routines to produce the print line that consists of the address of the data, the hex representation, and the EBCDIC translation of the data at that location.

Entry from: IEAVAD03, IEAVAD0B, IEAVAD0D via BALR.

Exit-normal: Back to caller.

Exit-error: Back to caller with a code of eight if the space for a save area is not available. An 0C4 abend occurs if an unexpected UPR happens.

IEAVAMSI — VIO Services Interface

Operation: This module manipulates page and external page tables and page frame table entries (PFTEs) for the virtual block processor (VBP) in support of VIO data sets.

Entry from: At IEAVAMSI from the Virtual Block Processor (VBP).

Exit to: VBP.

IEAVAR00 — RCT Initialization/Termination

Operation: This module performs address space initialization functions for a new address space and attaches the Dump Task and Started Task Control. When a task terminates, this module frees address space resources before termination and detaches the Dump Task and the Started Task Control task.

Entry from: At IEAVAR00 from IEAVEMIN.

Entry from: At IEAVAER0 from IEAVTAS1.

Exit to: To caller.

IEAVAR01 — RCT Common Processing

Operation: This modules routes control within the RCT modules to wait for a functional request in the Quiesce module and passes control to other routines to perform the requested functions. These functions are performed until a termination request is received, and then this module returns control.

Entry from: At IEAVAR01 from IEAVAR00.

Entry from: At IEAVARIA from IEAVTRT2.

Exit to: Caller.

IEAVAR02 — Quiesce Routine

Operation: This module waits for a functional request to be posted and determines the function requested. If posted by SRM, this module prepares an address for swap-out, or verifies that all tasks under the RCT are in long wait prior to returning. If posted for a Termination or Attention Exit request, this module returns control.

Entry from: At IEAVAR02 from IEAVAR01.

Entry from: At IEAVAFR2 from IEAVTRTS.

Exit to: Caller.

IEAVAR03 — Restore Routine

Operation: This module prepares an address space for execution after swap-in or handles Quiesce backout processing.

Entry from: At IEAVAR03 from IEAVAR01.

Entry from: At IEAVAR3A from IEAVTAS1.

Exit to: IEAVAR01.

Entry from: At IEAVAFR3 from IEAVTRTS.

Exit to: IEAVTRTS.

IEAVAR04 — Attention Exit Scheduler Routine

Operation: This module determines the attention level to be scheduled and schedules the associated attention exit.

Entry from: At IEAVAR04 from IEAVAR01.

Entry from: At IEAVAFR4 from IEAVTRTS.

Exit: To caller.

IEAVAR05 — Attention Exit Prolog Routine

Operation: This module acts as a front end of the user attention exit. It will issue TPUT or TGET to the terminal, create and initialize the TAIE, and will go to the attention exit in the user's key and state.

Entry from: At IEAVAR05 from IEAVEDS0.

Entry from: At IEAVART5 from IEAVTRT2.

Exit to: User attention exit or IEAVEOR.

Entry from: At IEAVAFR5 from IEAVTRTS.

Exit to: IEAVTRTS.

IEAVAR06 — Attention Exit Epilog Routine

Operation: This module acts as a resource manager for the TAXE queue. It frees the TAIE, restarts the subtasks of the TCB for which the attention exit has completed, performs housekeeping on the TAXE queue, and updates the available TAXE count.

Entry from: At IEAVAR06 from IEAVEOR.

Entry from: At IEAVAFR6 from IEAVTRTS.

Exit to: Caller.

IEAVAR07 — Attention Exit Purge Routine

Operation: This module acts as a resource manager for the TAXE queue. It is entered at task termination time so that inactive TAXEs for the terminating TCB can be dequeued. In addition to

the TAXE queue housekeeping, this module updates the available TAXE count.

Entry from: At IEAVAR07 from IEAVTSKT.

Entry from: At IEAVAFR7 from IEAVTRTS.

Exit to: Caller.

IEAVAX00 — STAX SVC Service Routine

Operation: This routine creates, maintains, and releases terminal attention exit elements (TAXEs) which contain information for scheduling terminal attention exits. These TAXEs are queued from the Region Control Task Data Area (RCTD) as they are created.

Entry from: At IGC0009F from IEAVESVC.

Exit to: IEAVEOR.

Entry from: At STXFRR from IEAVTRTS.

Exit to: IEAVTRTS.

IEAVBLDP — Build Quickcell Pool

Operation: This module creates, extends, or recovers (reformats) a quick-cell pool.

Entry: At IEABLDP.

Exit to: Caller.

IEAVCARR — Functional Recovery Routine for Module IEAVGCAS

Operation: This module attempts to clean up and recover from the following error situations encountered in module IEAVGCAS: program checks, machine checks, ABENDs, percolation from a subordinate FRR, and operator restarts.

Entry from: At IEAVCARR from IEAVTRTS.

Entry from: At IEAVTTRR from IEAVTRTS.

Entry from: At IEAVFARR from IEAVTRTS.

Exit to: IEAVTRTS.

IEAVCKEY — Change Storage Key

Operation: This module changes the storage key and fetch protection for virtual pages allocated from the problem program subpools.

Entry from: A supervisor state routine executing under protection key zero by branch entry only.

Exit to: Caller.

IEAVCKRR — Functional Recovery Routine for module IEAVCKEY

Operation: This module attempts to recover from program checks and machine checks encountered in module IEAVCKEY by restoring the original storage key of those pages changed.

Entry from: R/TM routine IEAVTRTS.

Exit to: R/TM routine IEAVTRTS.

IEAVCSEG — Create Segment

Operation: This module will perform one or two functions for one or more segments at a time: initialize a segment table entry and its associated page table, perform the preceding function and initialize the external page table associated with the page table plus initializing an entry in the SWAP control table (SPCT) for this segment.

Entry from: At IEAVCSEG from IEAVNP08 and IEAVGM00.

Entry from: At IEAVCSGB from IEAVEQR.

Exit to: Caller.

IEAVDELP — Delete Quickcell Pool

Operation: This module deletes all or part of a quickcell pool.

Entry: At IEAVDELP.

Exit to: Caller.

IEAVDLAS — Delete Address Space

Operation: This module makes available Real Storage Manager resources associated with a terminating address space. It also contains the SRB purge exit for all RSM SRBs.

Entry from: At IEAVDLAS from IEAVTERM.

Exit: To IEAVTERM.

Entry from: At IEAVSRBP from IEAVEPD0 or IEAVTMMT.

Exit to: Caller.

IEAVDSEG — Destroy Segment

Operation: This module destroys one segment table entry by invalidating it.

Entry from: At IEAVDSEG from IEAVRELS and IEAVEQR.

Exit to: Caller.

IEAVEAC0 — ASCB Dispatching Queue Manipulation Routine

Operation: This routine moves an ASCB to a new dispatching priority, adds an ASCB to a specified priority, or deletes a specified ASCB from the dispatching queue.

Entry from:
For entry from: Memory create/delete, SRM, and Swap out/in.
For entry IEAVEAC3: R/TM, for functional recovery routine processing.

Exit to: Caller.

IEAVEAT0 — ATTACH service routine

Operation: This module creates a new task as a subtask of an existing task. ATTACH creates a TCB and an SVRB.

Entry from: IGC00048 — SVC IH

Exit: IGC042R1 — Caller via Exit Prolog

IEAVECH0 — CHAP service routine

Operation: This module alters the dispatching priority of a task.

Entry from: IGC044 — SVC IH

Exit: IGC044 — Exit Prolog force dispatcher-entry IEAVEXPL.

IEAVEDR — Interprocessor Communication Direct Signal Routine

Operation: This module gives the interface needed to signal other CPU's which invoke any one of the twelve functions of: Sense, External Call, Emergency Signal, Start, Stop, Restart, Initial Program Reset, Program Reset, Stop and Store Status, Initial Microprogram Load, Initial CPU Reset, and CPU Reset.

Entry from: IEAVEDR via BALR.

Exit to: IEAVEDR for normal exit; and to a System Termination Facility for abnormal exit.

IEAVEDSR — Dispatcher Recovery

Operation: This module calls the necessary functions to clean up dispatching queues and the CPU in order to continue to dispatch ready work.

Entry from: IEAVEDSR.

Exit: To IEAVERTN, return address in IEAVESPR (super FRR).

IEAVEDS0 — Dispatcher

Operation: This module dispatches tasks, local supervisor routines and SRBs.

For entry: IEA0DS - main entry point from routines such as interrupt handlers, the exit prolog, and the lock manager.

For entry: IEAPDS2 - entered from the lock manager when suspending either a task which owns a local lock or an SRB.

For entry: IEAPDS6 - entered from EOT when it has deleted a task.

For entry: IEAPDS7 - entered by I/O and SVC FLIH on normal return to dispatcher.

For entry: IEAPDSRT - return point for all SRB routines.

Entry from: Supervisor routines.

Exit: via LPSW to dispatch: 1) SRB, 2) local supervisor routines, 3) a task, 4) the wait task.

For entry: DSJSTCSR - entered by dispatcher, lock manager's common suspend routine and IEAVRTI0 for job step timing calculations.

Exit: Return to caller.

IEAVEED0 — DETACH service routine

Operation: This module detaches a task and frees its resources such as the TCB and any associated program save areas.
For entry IGC062: Performs the DETACH operation.
For entry IGC062R1: Performs the DETACH operation.

Entry from:
For entry IGC062: SVC IH.
For entry IGC062R1: Supervisor routine via branch entry.

Exit to:
For entry IGC062: Caller via exit prolog.
For entry IGC062R1: Caller.

IEAVEEEP — IQE Purge Routine

Operation: Removes the IQEs for the terminating task from the asynchronous exit queue.

Entry from: End-of-task processing or ABEND processing.

Exit to: Return to caller.

IEAVEEER — Stage 3 Exit Effector Pecovery Routine

Operation: This module uses the queue verifier IEAVEQV0 to verify and correct the asynchronous exit queues.

Entry: IEAVEEER from Dispatcher recovery IEAVEDSR.

Exit: Via BR14.

IEAVEEE0 — Stage 3 Exit Effector

Operation: This module dequeues IQEs, RQEs, and SRBs from the asynchronous exit queues, and it also schedules the corresponding asynchronous exit by queueing the associated IRB to the TCB.

Entry point: IEA0EF03.

Entry from: The dispatcher.

Exit-normal: To caller via register 14.

IEAVEEE2 — Asynchronous Exit Queue Handler (Stage 2)

Operation: Places on the correct queue a request to schedule an asynchronous exit.

Entry from: A system routine (which passes an IQE, SRB, or an RQE to IEAVEEE2).

Exit to: Return to caller.

IEAVEES — Emergency Signal Second Level Interrupt Handler

Operation: This module routes control from the external FLIH to the appropriate receiving routine whenever the external FLIH receives an emergency signal interruption.

Entry from: IEAVEES via external FLIH BALRS.

Exit to: IEAVEEXT.

IEAVEEXP — Exit Prolog Routine

Operation: Performs exit functions for all SVC routines.

For entry IEAVEXPR: Provides normal SVC exit functions for all SVC routines if not the last RB on the TCB.

For entry IEAVEXP1: Provides SVC routines the ability to enter the dispatcher after exit processing.

For entry IEAVEXSV: ESR SVC routine entry.

Entry from:

For entry IEAVEXPR: All SVC routines.

For entry IEAVEXP1: All SVC routines.

For entry IEAVEXSV: Type 1 ESR.

Exit to:

For entry IEAVEXPR: Caller or dispatcher.

For entry IEAVEXP1: Caller or dispatcher.

For entry IEAVEXSV: Dispatcher.

IEAVEEXT — Interprocessor Communication External Interruption Routine

Operation: This interruption handler routine saves the status of the interrupted program, and decides what kind of external interruption occurred in order to give the proper SLIH control. On completion of the interruption process, it decides whether to return to the interrupted program or to the dispatcher.

Entry from: The interrupted program.

Exit to: The interrupted program or the dispatcher.

IEAVEE1R — External FLIH Recovery Routine 1

Operation: This module cleans up the external FLIH resources to allow ABEND to percolate the error.

Entry from: IEAVEE1R - from super FRR (IEAVESPR) if an error occurs during operation of the external FLIH.

Exit to: Via BR15 to IEAVEABD entry in IEAVESVR (common ABEND).

IEAVEE2R — External FLIH Recovery Routine 2

Operation: This module cleans up the FLIH resources to allow ABEND to percolate the error.

Entry from: IEAVEE2R - from super FRR (IEAVESPR) if an error occurs during operation of a recursive entry into the external FLIH.

Exit to: Via BR15 to IEAVEABD in IEAVESVR (common ABEND).

IEAVEE3R — External FLIH Recovery Routine 3

Operation: This module cleans up the external FLIH resources to allow ABEND to percolate the error.

Entry from: IEAVEE3R - from super FRR (IEAVESPR) if an error occured during the processing of a 2nd level of recursion into the external FLIH.

Exit to: Via BR15 to IEAVEABD in IEAVESVR (common ABEND).

IEAVEF00 — CIRB (Create IRB)

Operation: This module acquires space for an IRB, formats it according to the input parameters, and returns its address to the caller. It may also acquire space for a work area, which will be appended to the end of the IRB.

Entry from: SVC interruption handler (IEAVESVC)

Exit-normal: To caller via Br 14 if branch entry, or to exit prologue if SVC entry.

IEAVEIO — I/O First Level Interruption Handler

Operation: Saves status of interrupted program, invokes IOS to process interruption, and determines whether to return to the interruption program or to the dispatcher.

Entry from: From CPU following I/O interruption.

Exit to: Interrupted program (SRB routine) or to dispatcher.

IEAVEIOR — I/O FLIH Recovery Routine

Operation: This module makes the I/O FLIH recovery operable by clearing the I/O FLIH recursion indicator, allowing ABEND to percolate to one previous level of recovery.

Entry from: IEAVEIOR - super FRR (IEAVESPR).

Exit to: Via reg 15 to IEAVESVR at entry point IEAVEABD (common ABEND).

IEAVEIPR — IPC Functional Recovery Routine

Operation: This module clears RISGNL buffers and indicators so that RISGNL can be invoked again.

Entry from: IEAVTRTS.

Exit to: To R/TM via BR14.

IEAVELCR — Low Core Refresh

Operation: This module examines and sets correctly constants and addresses in low core.

Entry from: IEAVELCR.

Exit to: Via BR14 to SETLOCK verification (IEAVELKR).

IEAVELK — SETLOCK Service Routine

Operation: Sets and releases the various system locks by manipulating the lockwords associated with them. Locking synchronizes and controls access to the serially reusable resources of the system.

Entry from: A system routine that issued a SETLOCK macro instruction.

Exit to: Return to caller.

IEAVELKR — SETLOCK Repair Routine

Operation: This module corrects correlations between the locking hierarchy mask and the lock words. When a restart condition and a lock spin condition exists, it will clear spin lock words.

Entry from: IEAVELKR - to verify and reconstruct locks, lock table, and hierarchy mask. Entered from RTM before going to any FRRs.

Exit to: to ASVT verification routine (IEAVEVRR).

Entry from: IEAVLKRR - recovery routine for repair routine, determines if SETLOCK recovery is applicable, and if so, if a retry or percolation situation exists.

IEAVEMCR — Memory Create

Operation: This module establishes memory addressability, enqueues the new ASCB, and schedules a service request on the service priority list.

Entry from: IEDAY3 via XCTL, IEEVWAIT.

Exit to: Caller.

IEAVEMDL — Memory Delete

Operation: This module unassigns the ASID for a memory and notifies the system resource manager that a memory is deleted.

Entry from: IEAVTMMT.

Exit to: IEAVTMMT.

IEAVEMIN — Memory Initialization

Operation: This module creates a dispatchable TCB/SVRB on an ASCB's TCB ready queue. It also sets the SVRB to enter the program manager for linkage to the RCT.

Entry from: IEAVEMCR.

Exit to: IEAVAR00.

IEAVEMRQ — Memory Request

Operation: This module assigns an ASID for a new memory. It creates the ASCB and notifies the system resource manager that a new memory is being created.

Entry from: IEE0803D.

Exit to: IEE0803D.

IEAVEMS0 — Memory Switch Routine

Operation: Notifies the dispatcher that it should switch to another address space (memory) or begin searching for work on the ASCB dispatching queue.

Entry from: Supervisor routines that recognize more work has been made ready, i.e., IEAVELK or IEAVESC0.

Exit to: Return to caller.

IEAVENQ1 — Queue Manipulation Routine

Operation: This routine;

- Enqueues or dequeues a request for a serially reusable resource.
- Reserves and releases devices as requested by the RESERVE macro instruction.
- Performs "set must complete" or "reset must complete" operations or step-wide non-dispatchability.
- Purges queues at end-of-task or ABEND, as required.

For entry IGC048: Dequeue service routine.
For entry IGC056: Enqueue service routine.
For entry IEAVENQ2: ENQ resource manager.

Entry from:
For entry IGC048: SVC IH.
For entry IGC056: SVC IH.
For entry IEAVENQ2: R/TM via branch entry.

Exit to:
For entry IGC048: Exit prolog.
For entry IGC056: Exit prolog.
For entry IEAVENQ2: Caller.

IEAVEOR — Exit

Operation: This module terminates current RB or SPIE exit, in RB termination it also recognizes end-of-task and thereupon initiates and later terminates EOT processing.

Entry from: SVC

Exit to: Exit prolog via entry register 14; by invocation of normal task termination via issuance of SVC 13; to dispatcher IEAVEDS0.

IEAVEPC — Program Check Interruption Handler

Operation: The functions of this module are described under the entry point purposes below.
For entry IEAQPK00: This main entry point into the program FLIH decides what type of interruption has occurred and consequently decides which processor should be given control.

For entry IEAVPSRB: This routine performs the PIE/PICA processing necessary to satisfy an entry into a SPIE exit routine after a program check has occurred for an unlocked problem program TCB with a SPIE exit.
For entry IEASUSP: This entry to the paging supervisor saves the necessary status and performs operations necessary to suspend an SRB, a locked TCB, or an unlocked TCB.
For entry IEAVRSET: This entry to the paging supervisor causes the suspended TCB or SRB to be redispatched. If an error is indicated, R/TM is entered and will perform the necessary setup.

Exit to:

Exit-normal for IEAODS: This exit is from the program to the dispatcher if an SRB has been scheduled to perform PIE/PICA processing, or if a page exception has occurred and the paging supervisor indicates that I/O has to be done. This exit is to the dispatcher when a locally locked TCB has been suspended due to paging I/O. Exit form the program to a special entry in the dispatcher when an SRB has been suspended due to a page exception requiring paging I/O.
Exit-normal for R/TM (DATERR): Exit to R/TM when a translation specification has occurred and control register 0 bytes 8-12 are valid for translation, or segment exception recursion has occurred.
Exit-normal for RTM (PROGCK): Exit to R/TM under the following conditions:

- if a translation specification has occurred and control register 0 bytes 8-12 are invalid for translation,
- if a program recursion has occurred,
- if an interrupt other than an MC or PER has happened in the disabled storage,
- if a segment exception, invalid page exception, SSM instruction, or any program check has occurred in the SRB mode, in a locally locked TCB, in an unlocked TCB, in a supervisor state, or in an unlocked TCB in a problem state without a SPIE exit.

Exit-normal for interrupted program: Exit to the interrupted program via a LPSW if an MC or PER interrupt alone caused the program interruption or if a page exception has occurred and the paging supervisor was able to reclaim the page.
Exit-normal for SPIE routine: Exit to SPIE exit if a page exception has occurred and the interrupted program is neither in SRB mode nor in supervisor state and the TCBPIE17 bit is on Indicating.
Exit-normal for paging supervisor (RSM): Return to caller.
For exit-error: To load wait state.

IEAVEPCR — Program FLIH Recovery Routine

Operation: Recovery function:

- clears recursion indicators.
- allows percolation of recovery.

SRB function:

- Frees storage allocated to suspended SRB.

Entry from: FRR for recovery of program check FLIH. SRB suspended — from SETLOCK (IEAVELK) and program IH (IEAVEPPC)

IEAVEPDR — PURGEDQ Recovery Routine

Operation: This module performs FRR and ESTAE functions for PURGEDQ and cleans up for any errors during these function.

Entry from:

For entry IEAVEPDF: FRR for PURGEDQ handles errors during queue manipulation or scanning in PURGEDQ.
For entry: IEAVEPDE: ESTAE for PURGEDQ
For entry IEAVEPDS: retry routine for PURGEDQ.

Exit to:

For entry IEAVEPDE: R/TM via BALR.
For entry IEAVEPDF: R/TM via BALR.
For entry IEAVEPDS: PURGEDQ main entry point IGC123.

IEAVEPD0 — PURGEDQ Service Routine

Operation: Removes specified SRBs from the service manager queues. It ensures that the suspended SRBs have completed their processing before dequeueing them.

Entry from: A supervisor routine.

Exit to: Return to caller.

IEAVEQR — V = R Allocation

Operation: This module allocates and frees contiguous real storage for use in V=R regions.

Entry from: At IEAVEQR and IEAVEQRF from IEAVPRTO.

Exit to: IEAVPRTO.

Entry from: At IEAVEQRI from IEAVPFTE.

Exit to: IEAVPFTE.

Entry from: At IEAVEQRC by IEAVEQRI or IEAVSQA.

Exit to: IEAVEQR or IEAVSQA.

Error exit: To IEAVRCV.

Entry from: At IEAVEQRP for SRB page-out.

Exit to: Caller.

Error exit: To IEAVRCV or IEAVRMT.

IEAVEQV0 — Queue Verifier

Operation: The Queue Verifier (QV) verifies and corrects queue structures. It performs three basic functions:

- If the queue structure has been destroyed, QV will reconstruct the queue from any available information.
- Once the queue structure has been verified or reconstructed, QV will remove any elements that contain bad data.
- All errors detected, as well as corrective actions taken, are recorded in an output data area.

Entry from:
For entry IEAVEQV1: for single-threaded with header only (type 1).
For entry IEAVEQV2: for single-threaded with header and trailer (type 2).
For entry IEAVEQV3: for double-threaded (type 3).

Exit to: Return to caller.

IEAVERER — Restart IH Recovery Routine

Operation: This module makes the restart FLIH recovery function operable.

Entry from: IEAVERER - from super FRR (IEAVESPR) if error occurred during operation of the restart IH.

Exit to: BR15 to IEAVESVR at entry IEAVEABD to issue ABEND.

IEAVERES — Restart Interruption Handler

Operation: This entry saves the general and floating point registers and then proceeds to R/TM (restart).

Exit-normal: To R/TM.

IEAVERI — Interprocessor Communication Remote Immediate Service Routine

Operation: This module gives the interface needed to signal other CPU's which invoke a specific program on any of the online MP configured CPU's; it will also generate an emergency signal to the specified CPU which will then route control to the specified program.

Entry from: IEAVERI via BALR.

Exit to: Caller for normal exit.

IEAVERP — Interprocessor Communication Remote Pendable Service Routine

Operation: This module gives the necessary interface to signal other CPU's which invoke a specific program on any of the online MP configured CPU's; it will also generate an external call to the specified CPU which will then route control to the specific program.

Entry from: IEAVERP via BALR.

Exit to: Caller for normal exit.

IEAVESCR — Schedule Recovery Routine

Operation: This module:

1. verifies SRB journal queue and reschedules any remaining SRBs.
2. uses the queue verifier (IEAVEQV0) to verify the GSPL and LSPL queues for every address space in the system.

For entry IEAVESCR: To clean up for the schedule module IEAVESC0;
For entry IEAVESQV: For queue verification from the PURGEDQ FRR.

Entry from:
For entry IEAVESCR: Dispatcher recovery (IEAVEDSR).
For entry IEAVESQV: PURGEDQ FRR IEAVEPDR.

Exit to: either to Dispatcher recovery or the PURGEDQ FRR.
For entry IEAVESCR - Dispatcher recovery.
For entry IEAVESQV - PURGEDQ FRR.

IEAVESC0 — SCHEDULE Service Routine

Operation: Schedules SRBs by placing them on the correct queues (depending on priority). Also, alerts a CPU that it has work, invokes memory switch, and invokes the system resources manager when an address space is ready to be swapped in.

Entry from: Dispatcher (IEAVEDS0).

Exit to: Return to caller.

IEAVESPR — Supervisor Control FRR (Super FRR)

Operation: This module is a router routine for supervisor control. It determines retry and recovery addresses. In particular it can cause control to be routed to:

Function	Module name
RESTART FLIH recovery	IEAVERER
PROGRAM FLIH recovery	IEAVEPCR
EXTERNAL FLIH 1 recovery	IEAVEE1R
EXTERNAL FLIH 2 recovery	IEAVEE2R
EXTERNAL FLIH 3 recovery	IEAVEE3R
I/O FLIH recovery	IEAVEIOR
SVC FLIH recovery	IEAVESVR
DISPATCHER mainline	IEA0DS (DEFAULT)

Entry: IEAVESPR - for recovery control via LPSW. Entered from R/TM if an error occurred while one of the Supervisor Control FRR stacks was in control.

Entry: IEAVERTN - entry point for recovery routines called out of IEAVESPR mainline.

Exit to: R/TM, which will route control to one of the functions mentioned above.

IEAVESVC — SVC First Level Interruption Handler

Operation: This main entry into the SVC interruption handler decides whether the issuer can issue SVCs, if the issuer is authorized, and what type of SVC is asked for. It will also set up the proper input environment for the routine.

Entry from: Via a branch construction from the SVC interrupt handler mainline.

Exit-normal: R/TM (SVCERR).

Exit-normal: SVC routine.

Exit-error: SVC 13.

Exit-error: to IEAVEXPR (Exit prologue).

IEAVESVR —SVC IH Recovery Routine

Operation: This module performs two major functions:
1. it makes the SVC FLIH recovery operable by clearing the recursion indicator.
2. it issues ABENDs for SVC, external, I/O FLIH's.

Entry: IEAVESVR from super FRR (IEAVESPR) if error occurred during operation of SVC FLIH.

Exit to: falls through to entry IEAVEABD

Entry: IEAVEABD - from FLIH recovery routines for SVC (IEAVESVR same module) I/O (IEAVIOR and external...
1. issues ABEND macro to allow percolation of error.

Exit to: via SVC 13 (ABEND)

IEAVETCL — SUSPEND/RESUME/TCTL Processor

Operation: This module contains the routines that suspend a TCB/RB, resume a TCB/RB, and transfer control to a TCB/RB.

Entry from: Issuer of SUSPEND, RESUME, or TCTL macro.

Exit to: Issuer of SUSPEND or RESUME macro.

Exit to: TCB/RB requested on TCTL macro.

Exit to: SRB Exit if TCTL could not dispatch requested TCB/RB.

Error Exit: ABEND if SUSPEND RB=PREVIOUS was requested but no previous RB exists.

Error Exit: ABEND if TCTL or RESUME RETURN=NO issuer was in non-SRB mode.

IEAVEVAL — Validity Check

Operation: This module validates an address or address range by determining if the storage key and the protect key match.

Entry from: Supervisor routines via branch

Exit-normal: To caller.

Exit-error: To caller.

IEAVEVRR — ASVT Verification Reconstruction Routine

Operation: This module detects possible ASVT errors and initiates actions to correct them.

For entry IEAVEVRR: from IEAVELKR to detect errors in ASVT and initiate correction through FRR routine.

For entry IEAVVFRR: As an FRR from RTM to reconstruct the ASVT.

Exit to: to RTM.

IEAVEVT0 — EVENTS Service Routine

Operation: This module performs three services:

1. Event table creation and deletion, which causes an event table to be queued or dequeued from the requesting TCB.
2. ECB Initialization, which causes the ECB to be initialized to the event format, or if it is already posted, its address will be added to the event list of completed ECBs.
3. Event table wait, which causes the caller to stop processing and wait for the compeltion of one of n EVENTS.

For entry IEAVEVT0: Performs WAIT and/or ECB initialization.

For entry IGC125: Performs WAIT and/or ECB initialization.

For entry IEAVEVT1: Performs EVENT table creation or deletion.

Entry from:

For entry IEAVEVT0: Supervisor via branch entry.

For entry IGC125: SVC IH.

For entry IEAVEVT1: SVC IH through type-two extended SVC router.

Exit to:

For entry IEAVEVT0: Dispatcher (IEA0DS) or caller.

For entry IGC125: Exit Prolog or ABEND.

For entry IEAVEVT1: Exit Prolog or ABEND.

IEAVEXS — External Call Second Level Interrupt Handler

Operation: Upon an external call interruption, this module routes control from the external FLIH to the appropriate receiving routine.

Entry from: External FLIH BALRS 2, 10.

Exit-normal: To IEAVEXS.

IEAVFP — Find Page

Operation: This module returns the virtual addresses of the page and external page table entries corresponding to the input virtual address provided.

Entry from: At IEAVFP1 from non-RSM routines.

Exit to: Caller.

Entry from: At IEAVFP2 from other RSM routines.

Exit to: Caller.

IEAVFRCL — FREECELL Routine

Operation: This module returns a cell to quickcell pool.

Entry from: At IEAVFRCL.

Exit to: Caller.

IEAVFREE — PGFREE

Operation: This module executes a PGFREE request or backs out a partially completed PGFIX request that is being cancelled.

Entry from: At IEAVFREE from IEAVFXLD or IEAVPSI.

Exit to: Caller.

IEAVFXLD — Page Fix/Page Load

Operation: This module brings virtual storage pages into real storage if they are not already in, and for PGFIX requests, fixes these pages in real storage such that no paging I/O will be performed for the pages.

Entry from: At IEAVFXLD from IEAVPSI.

Exit to: IEAVPSI.

Entry from: At IEAVFXL from any RSM routine that decrements the PCB count in a PGFIX or PGLOAD root PCB to zero.

Exit to: Caller.

IEAVGCAS — Create/Free Address Space Routine, VSM Task Termination Routine

Operation: This routine establishes addressability for a new address space, frees task-related storage at task termination and frees VSM global storage at address space termination.

Entry from: At IEAVGCAS from IEAVEMCR routine.

Entry from: At IEAQSPET from IEAVEOR.

Entry from: At IEAVGFAS from IEAVTSKT routine.

Exit to: Return to caller.

IEAVGFA — General Frame Allocation

Operation: This module performs real frame allocation services, including reclamation. Paging I/O is started where necessary to retrieve an auxilary storage copy of a page.

Entry from: At IEAVGFA from IEAVPIX, IEAVFXLD, or IEAVSWIN.

Exit to: Caller.

Entry from: At IEAVGFAD from IEAVPFTE via IEAVEDS0.

Exit to: IEAVEDS0.

Error exit: To IEAVRCV.

IEAVGFRR — Functional Recovery Routine for GETMAIN/FREEMAIN

Operation: This module recovers the function of GETMAIN/FREEMAIN after a machine or system error has occurred.

Entry from: At IEAVGFRR from IEAVTRTS.

Exit to: IEAVTRTS.

IEAVGM00 — GETMAIN/FREEMAIN Service Routine

Operation: This module allocates (GETMAIN) and deallocates (FREEMAIN) virtual storage in the SWA and CSA and in the LSQA, SWA, and user region of each address space. A secondary function in support of system management facilities (SMF) is to produce actual storage-used values.

Entry from: At IGC004 from IEAVESVC for storage type GETMAIN.

Entry from: At IGC005 from IEAVESVC for register type, unconditional GETMAIN or FREEMAIN.

Entry from: At CSSRANCH for register type GETMAIN or FREEMAIN of a 16 byte cell from VSM control block cell pool.

Entry from: At IGC010 from IEAVESVC for register type, unconditional GETMAIN or FREEMAIN.

Entry from: At IGC120 from IEAVESVC for register type GETMAIN or FREEMAIN.

Entry from: At GMBRANCH for storage type GETMAIN.

Entry from: At FMBRANCH for storage type FREEMAIN.

Entry from: At RMBRANCH for register type, unconditional GETMAIN or FREEMAIN.

Entry from: At CRBRANCH for register type GETMAIN or FREEMAIN of a 16 byte cell from VSM control block cell pool.

Entry from: At GLBRANCH for register type GETMAIN or FREEMAIN of global subpools without local lock.

Entry from: At GETMAING from IEAVPRT0.

Entry from: At MRELEASF from IEAVPRT0.

Entry from: At MRELEASR from IEAVRELS.

Exit to: Caller.

IEAVGPRR — GETPART/FREEPART Functional Recovery Routine

Operation: This module attempts to clean up and recover from the following error situations encountered in module IEAVPRT0; program checks,

machine checks, ABENDs, percolation from a subordinate FRR, and operator restart.

Entry from: At IEAVGPRR from IEAVTRTS.

Exit to: IEAVTRTS.

Entry from: At PRTOERTN from IEAVSY50 via IEAVEDS0.

Exit to: IEAVEDS0.

IEAVGTCL — GETCELL Routine

Operation: This module allocates a cell from an existing quickcell pool.

Entry from: At IEAVGTCL.

Exit to: Caller.

IEAVID00

Entry from: IGC041.

Operation: This module performs two functions.
To identify an embedded entry to the system.
To identify to the system a single copy module loaded in subpool 0.

Entry from: SVC interruption handler.

Exit to: Exit Prolog
FRRSUC41: Same as IEAVLK03 FRR for identify

IEAVINV — Page Invalidation Routine

Operation: The routine is divided into two parts known as the Master and the Slave subroutines. The master synchronizes all online CPUs, sets the invalid bit in the PGTE, purges the translation look-aside buffer (TLB) on the Master CPU, and signals the other CPUs to purge their TLBs. The slave portion purges the TLB on its CPU when the master signals it to do so.

Entry from: At IEAVINV from an RSM routine.

Exit to: Caller.

Entry from: At IEAVINVA from IEAVEES.

Exit to: IEAVEES.

IEAVIOCP — Paging I/O Completion Processor

Operation: This module performs the address-space-dependent processing required to indicate that a page-in operation has completed. The only PCBs processed are these with I/O complete and for the address space IEAVIOCP is running in.

Entry from: At IEAVIOCP from IEAVEDSO.

Entry from: At IEAVCPBR.

Exit to: Caller.

IEAVITAS — Initialize Address Space

Operation: This module builds and initializes RSM control blocks required to define an address space.

Entry from: At IEAVITAS from IEAVGCAS.

Exit to: IEAVGCAS.

IEAVLK00 — Program Management Module 1 of 2

Operation: This module contains the description and code for several service routines:

- Link SVC 6 entered at IGC006,
- XCTL SVC 7 entered at IGC007,
- Load SVC 8 entered at IGC008,
- Delete SVC 9 entered at IGC009,
- Synch SVC 12 entered at IGC012.

In addition to the above service routines, this module includes other code whose functions are described below.

For entry IGC006: The functions of IGC006 are the following:

- performs a validity check of resources,
- establishes linkage to the module,
- searches for and determines the load module to which linkage is desired,
- creates a program request block and creates a contents directory entry under certain conditions,
- carries modules to be fetched,
- performs a sharing of load modules in accordance with reusability attributes,
- invokes the test module via an SVC call in IEAVLK01 for TSO modules in test.

For entry IGC007: The functions of entry IGC007 are the following:

- performs linkage to the load module specified,
- searches lists and libraries,
- creates and updates PRB and CDE as for IGC006,
- fetches and packages as done for Link,
- reclaims the current SVRB for Type 4 SVC's during the processing.

For entry IGC008: The functions of entry IGC008 are the following:

- acquires a specified load module and retains the module for use by the task issuing the load,
- searches as in IGC006,
- creates a twelve-byte load list element when a module is retained by a task,
- increases the use count in the CDE when a module is loaded,
- keeps a responsibility count in the LLE,
- will not queue load requests for a serially reusable module; it returns entry point even if module is in use;
- causes modules to be fetched as for IGC006.

For entry IGC009: The functions of entry IGC009 are the following:

- indicates that the usage of a module is complete,
- locates the named module,
- decreases the use count in the CDE under certain conditions and if the count goes to zero, it forces the purging of the module.

For entry IGC009: The functions of entry IGC012 are the following:

- Allows the supervisor routine to give control to a problem program routine and to regain control in supervisor mode.
- Gives task recovery the ability to schedule a synchronous exit routine.

For entry IEAQCS01: This entry provides supervisor-assisted linkage for ATTACH requests.
For entry IEAQCS02: This entry starts a queued request through mainline LINK.
For entry IEAQCS03: This entry restarts a queued serially reuseable request.
For entry IEAVVMSR: This entry searches the pageable LPA directory for an LPDE representing the requested module.
For entry IEAQCDSR: This entry searches a CDE queue for a requested name.
For entry IEAQCS04: This entry address is the start of a list of addresses which IEAVLK01 uses to return to IEAVLK00. There is no linkage and this is an external reference and not executable.
For entry CDLKBASE: This entry address resets the base address for IEAVLK00 on return from IEAVLK01. There is no linkage as this is an external reference and not executable.

Entry from:

For entry IGC006: SVC IH.
For entry IGC007: SVC IH.
For entry IGC008: SVC IH.
For entry IGC009: SVC IH.
For entry IGC012: SVC IH.
For entry IEAQCS01: ATTACH (IEAVEAT0) or memory create.
For entry IEAQCS02: POST (IEAVSY50.
For entry IEAQCS03: POST (IEAVSY50.
For entry IEAVVMSR: Control program and user routines.
For entry IEAQCDSR: Control program and user routines.

Exit to:

For entry IGC006: Routine to be linked via EXf; WAIT if request was queued; or ABEND for error.
For entry IGC007: Routine to be linked via EXIT; WAIT if request was queued; or ABEND for error.
For entry IGC008: Caller via exit prolog; or ABEND for error was queued; or ABEND for error.
For entry IGC009: Caller.
For entry IGC012: Entry point in register 15 via EXIT.
For entry IEAQCS02: Same as IGC006.
For entry IEAQCS02: Same as IGC006.
For entry IEAQSC03: Same as IGC006.
For entry IEAVVMSR: Return on register 14+0 if LPDE found; return on register 14+4 if LPDE not found.
For entry IEAQCDSR: Return on register 14+0 if CDE found; return on register 14+4 if CDE not found.

IEAVLK01 — Program Management Module 2 of 3

Operation: This module performs the following services for IEAVLK00:

- controls the order of queue and library searches,
- interfaces with BLDL service routine,
- interfaces with program fetch service routine,

- provides a switch for DSS,
- provides an interface to test,
- processes alias PDS entries so that the major name module can be reused, if possible, for an alias request.

Entry from: IEAVLK00 via branch.

Exit-normal: To IEAVLK00.

IEAVLK02 — Program Management Resource Management Routines

Operation: This module cleans up CDEs and any associated storage at EXIT, DELETE, ABEND and EOT. It also prevents unwanted ABENDs in Program Management by intercepting them and correcting errors if possible.

Entry from: ABEND.

Exit to: Program management.

Entry Point: CDHKEEP: This entry releases the module, extent list, major CDE, and associated minors as required.

Entry from: DELETE Service Routine.

Exit to: Return to caller via BR 14.

Entry Point: IEAPPGMX: This routine frees the Program Management resources at EXIT and determines if a task termination is in progress, and if so, frees any LLE's left on TCBLLE queue, their associated CDE's and its associated storage.

Entry from: EXIT via branch with LOCAL lock held.

Exit to: Caller (SVC 3) via branch.

Entry Point IEAPPGMA: This routine removes partially loaded programs during ABEND processing.

Entry from: RTM1 via branch.

Exit to: Return to caller via BR 14.

IEAVLK03

Entry from: FRRPGMMG
Program Manager Recovery FRRPGMMG routines for the basic program management functions and for the FRRPGMMX exit resource manager for program management.

Entry from: R/TM.

Exit to: R/TM.

IEAVMASV: Communication Task Asynchronous Service Routine

Operation: The asynchronous service routine performs TPUT services for the communication task. The TPUT services issue the WAIT macro instruction.

Entry from: IEAVTPUT and IEAVMASV.

Exit to: A branch return to the dispatcher based on the address in register 14.

IEAVMDOM: Communication Task DOM Processor

Operation: The communication task DOM processing routine processes the DOM control blocks (DOMCs) created by the DOM macro instruction processing routine (IEAVXDOM). For each DOMC message identification, this routine attempts to eliminate the WQE that has that message identification and the ORE associated with that WQE, if an ORE exists. The device support processor is then called to eliminate graphic messages with the same message identification.

Entry from: IEAVMDOM.

Exit to: A branch return to the communication task's wait service routine based on the address in register 14.

IEAVMDSV: Communication Task Device Service Routine

Operation: The device service routine prepares the communication task for and issues SVC 72 — the device support processor (DSP). The preparation includes the handling of open pending, close pending, I/O completion interrupt, and attention interrupt pending. This routine also handles some clean up of the write queue elements (WQEs), operator reply elements (OREs), and console queue elements (CQEs) when these control blocks are no longer needed.

Entry from: IEAVMDSV.

Exit to: A branch return to the module that called this module based on the address in register 14.

IEAVMED2 — WQE/ORE Purge Routine

Operation: This module pruges WTO elements and ORE (operator reply elements) at task or memory termination. In particular, it:

- Frees any write wait blocks (WWBs) from the ORE and WQE chain.
- Marks any OREs to indicate that deletion is in progress.
- Ends any multi-line WQEs for this task or memory and marks them as 'not-waiting'.
- Creates a DOMCB (delete operator message control block) for any descriptor code 7 (end of job) messages in this task or memory.
- Creates a DOMCB for a DOM by ASID (address space identifier) if this is a memory deletion.
- If this is a job step deletion, it creates a DOMCB for a DOM by ASID and jobstep TCB.
- Marks any active graphic device UCME to indicate there is 'DOM' work to be done.
- Stops monitor requests for this task or memory (by calling the monitor control routine, IEAVMNTR.)
- Posts the UCMDECB
- Passes any task-related DOMCBs to the subsystem interface.

Entry from: Memory or task termination.

Exit: To caller.

IEAVMFRR: Communication Task Functional Recovery Routine (FRR) or ESTAE Controller

Operation: The FRR/ESTAE controller receives control from recovery termination management (RTM). The controller processes both FRR and ESTAE attempts to recover from abnormal terminations in the communication task.

Entry from: IEAVMFRR and IEAVMEST.

Exit to: A branch return to the recovery termination management (RTM) routine based on the address in register 14.

IEAVMNTR — Monitor Queue Manager

Operation: This module maintains the monitoring status of operators' consoles and TSO terminals. This is done primarily for the MONITOR and STOPMN command processors.

Entry from: IEAVMED2; IEAVSWCH; IEE5503D; IEE7103D; IKJEFF00; IEE4603D.

Exit to: Caller.

IEAVMODE — Mode Change Routine

Operation: This module:

1) changes key in SVCOPSW (SVC entry),
2) changes state in SVCOPSW (SVC entry), or
3) changes system mask (SVC entry or branch entry).

1. IGC107
This routine changes the state of the system through alterations to the RB old PSW.

Entry from: SVC IH.

Exit to; Exit prologue.

IEAVMQR0 — Communication Task Process Message for Inactive Terminal (Console)

Operation: The processs message for inactive console routine gives the master console operator an opportunity to delete, queue to the master console, or continue queueing messages that are sent to a specific console by identification.

Entry from: IEAVMWSV.

Exit to: A branch return to the dispatcher based on the address in register 14.

IEAVMQWR: Communication Task Wait Service Routine

Operation: The wait service routine waits on all of the event control blocks (ECB) that are posted for the communication task.

Entry from: IEAVMQWR.

Exit to: No exit; this is a never-ending routine.

IEAVMWSV — Communication Task Console Queueing Routine

Operation: The console queueing routine selectively queues messages to console output queues.

Entry from: IEAVMWSV, IEECMENQ, and IEECMQCN.

Exit to: A branch to the device service routine (IEAVMDSV). The device service routine will return control to the communication task's wait service routine (IEAVMQWR).

IEAVMWTO — Communication Task Multiple-Line WTO Service Routine

Operation: The communication task multiple-line WTO (MLWTO) service routine prepares and chains major and minor write queue elements (WQEs) associated with multiple line messages.

Entry from: IEAVMWTO.

Exit to: A branch return to the WTO and WTOR macro instruction processing routine (IEAVVWTO) based on the address in register 14.

IEAVOUT — Page Out

Operation: This module processes all page-out requests initiated by the PGOUT macro instruction. If the page needs to be written to auxiliary storage, the I/O request will be started by this module.

Entry from: At IEAVOUT from IEAVPSI.

Exit to: IEAVPSI.

IEAVPCB — PCB Manager

Operation: This module moves PCBs to various queues as requested and ensures that an adequate supply of PCBs are in the free queue pool for requesters. The SRB pool is also maintained.

Entry from: At IEAVPCB from an RSM module or from IEAVNIP0.

Exit to: Caller.

IEAVPFTE — PFTE Enqueue/Dequeue Routine

Operation: This module moves PFTEs to and from PFTE queues as directed or needed. It also restarts deferred GFA requests and monitors the level of the AFQ (available frame queue).

Entry from: At IEAVPFTE from RSM modules.

Exit to: Caller.

IEAVPIOI — Swap I/O Initiator

Operation: This module invokes the ASM interface routine to initiate paging I/O for swap-out of an address space.

Entry from: At IEAVPIOI from IEAVEDS0.

Exit to: IEAVEDS0.

IEAVPIOP — Page I/O Post

Operation: This module performs functions relating to paging I/O, including frame validation and freeing of PCBs and frames.

Entry from: At IEAVPIOP from ILRI0C00, ILRMON00, or ILRQIO00.

Exit to: Caller.

Entry from: At IEAVOPBR from IEAVIOCP, IEAVTERM, IEAVRELS, IEAVAMSI, IEAVGFA, or IEAVSOUT.

Exit to: Caller.

IEAVPIX — Program Check Interrupt Extension

Operation: This module functions as an interface routine between the Program Check Interruption Handler (PCIH) and the General Frame Allocation (GFA) routines.

Entry from: At IEAVPIX from IEAVEPC.

Exit to: IEAVEPC.

IEAVPREF — Preferred Area Allocation Steal

Operation: This module steals a frame in the preferred area to be used for SQA, LSQA, or for a long-term fixed page of a nonswappable address space.

Entry from: IEAVGFA or IEAVSQA.

Exit to: Caller.

IEAVPRT0 — GETPART/FREEPART Routine

Operation: This module allocates or frees a virtual region.

Entry from: At IEAVPRT0 from IEAVGM00.

Exit to: IEAVGM00 or to IEAVEOR.

Error Exit to: IEAVTRT2.

IEAVPSI — Page Services Interface (PSI) Module

Operation: This module receives all page service requests. The requests serviced are PGFIX, PGFREE, PGLOAD, PGRLSE and PGOUT. This module checks input parameters for errors and routes control between the page services modules necessary to satisfy the input request.

Entry Point: At IGC112 from IEAVESVC.

Exit to: IEAVEOR.

Error exit: To IEAVRCV.

Entry from: At IGC113 from IEAVESVC.

Exit to: IEAVEOR.

Error exit: to IEAVRCV.

Entry from: At IEAVPSII from another RSM routine.

Exit to: Caller.

Error exit: To IEAVRCV.

Entry from: at IEAVPSIB from a Supervisor State Key 0 routine

Exit to: Caller.

Entry from: at NEXTVSL from another RSM routine.

Exit to: Caller.

Entry from: At IEAVPSIX from IEAVTCCW.

Exit to: IEAVTCCW

Error exit: To IEAVRCV.

Entry from: At IEAVPSIF from IEAVTCCW.

Exit to: IEAVTCCW.

Error Exit: To IEAVRCV.

IEAVRCF — Real Storage Reconfiguration Routine

Operation: This module attempts hardware recovery (storage data and key errors), fields requests for varying storage offline and online, and provides current frame status.

Entry from: At IEAVRCF from IEEMPVST

Entry from: At IEAVRCFC from IEAVRCF via IEAVEDS0

Exit to: Caller.

Entry from: At IEAVRCFI from IEAVPFTE.

Exit to: IEAVPFTE.

IEAVRCV — Real Storage Management Recovery

Operation: This module attempts to recovery from program checks, machine checks, operator restarts, percolation from subordinate FRRs, and expected and unexpected ABEND requests in RSM routines.

Entry from: At IEAVRCV from IEAVTRTS.

Exit to: IEAVTRTS.

Entry from: At IEAVRCV2 from IEAVTRTS for errors in IEAVRCV.

Exit to: IEAVTRTS.

IEAVRELS — Page Release (PGRLSE)

Operation: This module frees real and auxiliary storage associated with a virtual page, quiescing any outstanding paging I/O for the page at the same time.

Entry from: At IEAVRELS from IEAVPSI.

Exit to: IEAVPSI.

Entry from: At IEAVRELV from IEAVGM00.

Exit to: IEAVGM00.

Entry from: At IEAVRELF from IEAVIOCP, IEAVFREE, or IEAVTERM.

Exit to: Caller.

IEAVRFR — Real Frame Replacement

Operation: The Select routine is invoked by the SRM when the steal algorithm to be executed. It is passed a criteria number by which it is to decide which frames are eligible to be stolen.
The Steal routine is used to remove pages from a given address space and return the frames to the available pool.

Entry from: At IEAVRFR from IRARMSRV.

Exit to: IRARMSRV.

Entry from: At IEAVFRFA from IEAVEDS0.

Entry from: At FREEPAGE from either Steal or Select.

Exit to: Caller.

IEAVRTI0 — Timer Second Level Interrupt Handler

Operation: This module processes interrupts caused by a synchronization check, by the CPU timer, or by the clock comparator. This module also contains the TQE enqueue and dequeue routines.

Entry from: At IEA0TI00 from IEAVEXS.

Exit to: IEAVEXS.

Entry from: At IEAQTE00.

Exit to: Caller.

Entry from: At IEAQTD00.

Exit to: Caller.

Entry from: At IEAVRCKQ.

Exit to: Caller.

Entry from: At IEAVRQCK from IEAVEXS.

Exit to: IEAVEXS.

Entry from: At IEAVRSAE from IEAVEDS0.

Exit to: IEAVEDS0.

IEAVRTI1 — Timer Purge and Recovery

Operation: This module contains four functions: the TQE Purge function called by a terminating task or memory; the hardware recovery routine called by R/TM; initializing timer control block fields and timer hardware when a new CPU is varied online; and the FRR for timer supervision. Included in this function is a TQE validation routine.

Entry from: At IEAQPGTM from IEAVTSKT or IEAVTMMT.

Entry from: At IEAVRSPG from IEAVEDS0.

Entry from: At IEAVRCLS from IEAVTRTH.

Entry from: At IEAVRCLX from IEAVEDS0.

Entry from: At IEAVRNEW from IEEVCPU.

Entry from: At IEAVRSPN from IEAVEDS0.

Entry from: At IEAVRFRR from IEAVTRTS.

Entry from: At IEAVRTVR from IEAVRFRR or IEAVEQV0.

Exit to: Caller.

IEAVRTOD — TOD Clock Manager

Operation: This module initializes TOD clocks at IPL and when a CPU is varied online and resets or resynchronizes TOD clocks suffering machine checks.

Entry from: At IEAVRINT from IEEVIPL.

Entry from: At IEAVRSSC from IEAVRTOD and IEEVCPU.

Entry from: At IEAVRCLA from IEAVEDS0.

Entry from: At IEAVRNOT from IEEVCPU.

Entry from: At IEAVRCAN from IEEVCPU.

Exit to: Caller.

IEAVRT00 — TTIMER Service Routine (SVC 46) and STIMER Service Routine (SVC 47)

Operation: TTIMER will supply the time remaining in a task's time interval which was previously established via the STIMER routine. TTIMER will also, if requested, cancel the interval. STIMER establishes a time interval for the requesting task.

Entry from: At IGC0004F from IEAVESVC.

Entry from: At IGC0004G from IEAVESVC.

Exit to: IEAVEOR.

Error exit: To IEAVTRT2.

IEAVRT01 — TIME Service Routine (SVC 11)

Operation: This module supplies the user with local time and date, or optionally, with Greenwich Mean Time and date. The current value of the Time-of-Day clock can also be requested.

Entry from: At IGC0001A from IEAVESVC.

Exit to: IEAVEOR.

Entry from: At IEAVRTME from IEAVEES.

Exit to: IEAVEES.

Entry from: At TIMESTAE from IEAVTAS1.

Error exit: To IEAVTRT2.

IEAVRT02 — SETDIE Routine

Operation: This module allows a system program to establish a real-time interval. After the interval has passed, the program's disabled interrupt exit (DIE) routine gets control.

Entry from: At IEAVRDIE from system caller.

Exit to: System caller.

IEAVSETS — STATUS service routine

Operation: The module manipulates TCB indicators to change the dispatchability of tasks, SRBs, or the system.

For entry IGC079: Manipulates dispatchability of tasks, SRBs, or the system.
For entry IGC07902 Manipulates the dispatchability of tasks and SRBs.
For entry IGC07903: Starts SRB in swapped-in address spaces.
For entry IEAVSSNQ: Stops non-quiesceable SRBs in address spacess being swapped-out.
For entry IEAVESSS: Completes stop SYNCH processing.
For entry IEATRSCN: Searches the subtask tree structure.

Entry from:

For entry IGC079: SVC IH.
For entry IGC07902: Supervisor routines via branch entry.
For entry IGC07903: Swap-in.
For entry IEAVSSNQ Swap-out.
For entry IEAVESSS: EXIT and exit prolog.
For entry IEATRSCN: Branch entry from system routines.

Exit to:

For entry IGC079: Exit prolog.
For entry IGC07902: Caller or dispatcher.
For entry IGC07903: Caller or dispatcher.
For entry IEAVSSNQ: Caller or dispatcher.
For entry IEAVESSS: Caller or dispatcher.
For entry IEATRSCN: Branch to caller via BR 11 if no task found, or BR 14 if task is found.

IEAVSOUT — Swap Out

Operation: This routine proposes to process or move an address space from the system at the direction of the System Resource Manager (SRM). This includes purging all paging I/O for the address space, purging fixes for the address space, freeing LSQA frames, indicating which pages should be swapped in, and building the control blocks which will control the swap-out of pages which need to be written to auxiliary storage.

Entry from: IEAVSOUT from IEAVAR02.

Exit to: IEAVAR02.

IEAVSQA — LSQA or SQA Allocation

Operation: This module allocates real storage frames for use as a SQA or LSQA pages.

Entry from: At IEAVSQA from IEAVGM00 or IEAVITAS.

Exit to: Caller.

IEAVSTAA: Communication Task STAR Routine

Operation: The STAR routine is the last attempt to recover from an abnormal termination in the communication task. Previous attempts to recover may have been made by the functional recovery routine (FRR) and ESTAE processing (IEAVMFRR0). The STAR routine reinitializes all fields that are necessary to make the system believe the communication task has not failed.

Entry from: IEAVSTAA.

Exit to: A branch return to the recovery termination management routine (RTM) based on the address in register 14.

Error exit: A branch return to the recovery termination management routine (RTM) based on the address in register 14.

IEAVSTA0 — (E)STAE Service Routine (SVC 60)

Operation: This module creates, overlays, cancels, or propagates STAE control block(s).

Entry from: SVC 60 call or a branch entry by another SVC routine.

Exit-normal: To caller via exit if SVC entered, or directly to caller if branch entered.

Exit-error: ABEND caller.

IEAVSWCH: Communication Task Console Switch Routine

Operation: The console switch routine analyses and switches console functions.

Entry from: IGC0407B.

Exit to: A branch return to the communication task's wait service routine (IEAVMQWR) based on the address in register 14.

IEAVSWIN — Swap In Processor

Operation: This module performs the processing required to initiate a swap-in of an address space from the auxiliary paging space. The secondary entry performs processing to connect the address space to the set of active address spaces when the required I/O is complete.

Entry from: At IEAVSWIN from IEAVEDS0.

Exit to: IEAVEDS0.

Entry from: At IEAVSIRT from IEAVPIOP, IEAVGFA, IEAVIOCP or IEAVTERM.

Exit to: Caller.

IEAVSWPC — Swap-out Completion Processor

Operation: This routine frees the frames and PCBs for swap-out and removes the address spaces from the dispatching queue.

Entry from: ILRPAGCM.

Exit to: Caller.

IEAVSY50 — Wait/Post Service Routine

Operation: This module performs two services:

- WAIT, which causes the caller to stop processing and wait for the occurrance of one or more events.
- POST - signals completion of an event in an ECB and determines if the waiting task has become ready.
  - identifies and deletes user POST exit routines and routes control to them.

For entry IGC001: Performs the WAIT service.
For entry IEAVWAIT: Performs the WAIT service.
For entry IGC002: Performs the POST service.
For entry IEA0PT01: Performs the POST service.
For entry IEA0PT02: Performs the POST service.
For entry IEA0PT03: Performs the POST service for programs executing as a POST exit routine.
For entry IEA0PT0E: Performs the POST exit identification and deletion service.
For entry IEARPOST: POST resource manager.

Entry from:
For entry IGC001: SVC IH.
For entry IEAVWAIT: Supervisor routine via branch entry.
For entry IGC002: SVC IH.
For entry IEA0PT01: Supervisor routine via branch entry.
For entry IEA0PT02: Supervisor routine via branch entry.
For entry IEARPOST: R/TM.

Exit to:
For entry IGC001: Exit prolog (IEAVEXP1), or ABEND.

For entry IEAVWAIT: Dispatcher (IEA0DS).
For entry IGC002: Exit prolog or ABEND.
For entry IEA0PT01: Caller.
For entry IEA0PT02: Caller.
For entry IEARPOST: Caller.

IEAVTABD — ABDUMP

Operation: This module manages the SYSABEND AND SYSUDUMP data sets. It provides the interface to SNAP for dump requests to one of these data sets during recovery/termination processing.

Entry from: IEAVTRTC via BALR.

Exit-normal: Return to IEAVTRTC.

Exit-error: Return to IEAVTRTC.

IEAVTABI — ABDUMP Initialization

Operation: This module reads PARMLIB IEAABD00 and IEADMP00 and sets the requested dump options in the Recovery/Termination Control Table for later use by ABDUMP (IEAVTABD).

Entry from: IEAVNPAG via BALR.

Exit-normal/error: Return to IEAVNPAG.

IEAVTACR — Alternate CPU Recovery

Operation: This module uses a 'good' (recovery) CPU to take over the work in progress on a 'dead' (malfunctioning) CPU at the time of its failure, and to supervise the cleanup of that work until the system can resume normal operation:

Entry from:

a from RSM EMS/MFA SLIH on the recovery CPU
b from Lock manager and certain global spinners when conflict is detected between recovery and dead CPUS work.
c from Dispatcher when recovery or dead CPU work is interrupted.

Exit:

a To caller via reg. 2.
b & (c) To interrupted work of other CPU. If this is first time for dead CPU work exit to RTM for FRR recovery.
c If work of each CPU is in enabled condition exit to dispatcher.

IEAVTAS1 — Task Recovery Pre-exit Processing

Operation: This module sets up workareas for ESTA exit, gives the exit control and calls IEAVTAS2 and IEAVTAS3 for post-exit processing. The pre-exit functions of this module are as follows:

- identify the SCB (ESTAE/STAE exit)
- obtain and initialize SDWA
- perform the user I/O options requested on (E)STAE macro
- set up interface for the user exit routine
- issue SYNCH macro to pass control to the user exit routine

Entry from: IEAVRTC via BALR.

Exit-normal: Return to IEAVTRTC.

IEAVTAS2 — Task Recovery Post Exit Processing

Operation: This module performs the requests made by the user exit routine. Its functions are the following:

- performs a validity check on the SDWA
- track SDWA by issuing HOOK macro
- perform recording if requested
- copy dump options from SDWA
- suppress STAI percolation, if requested
- determine whether to retry or continue with termination

Entry from: IEAVTAS1 via BALR.

Exit-normal: IEAVTAS1.

IEAVTAS3 — Setup for User Retry or Continue with Termination

Operation: This module performs processing for user retry or percolation. Processing consists of the following:

Retry:

- identify the RB for retry
- initialize or freemain SDWA for retry
- close embedded DCB-s
- purge paging I/O
- purge outstanding WTOR requests for STAE retry only
- purge entries in type 1 message table
- purge enqueued resources if ESTAR
- purge TQEs if ESTAR
- set pointer to EXIT (RB) or exit prolog (SVRB) in resource PSW
- perform SCB clean-up
- handle pseudo retry into IEAVTAS1 if there is recursion

Percolation: (continue with termination)

- free SDWA

Entry from: IEAVTAS1 via BALR

Exit-normal: IEAVTAS1.

IEAVTB00 — SPIE/EXTRACT service routine

Operation: The module provides two services:

- SPIE, which provides the problem program with a means of specifying an error exit

routine in response to one or more program error interruptions.
- EXTRACT, which provides a means of obtaining the address of the values of a set of fields without going throught the system.

For entry IGC0001D: Performs SPIE mainline processing.
For entry IEAVSPIE: SPIE resource manager.
For entry IEAVSPI: Checkpoint/restart entry.
For entry IGC00040: Performs EXTRACT mainline processing.
For entry IGC00040+8: Performs EXTRACT mainline processing.

Entry from:
For entry IGC0001D: SVC IH.
For entry IEAVSPIE: R/TM.
For entry IEAVSPI: Checkpoint/restart.
For entry IGC00040+8: Supervisor routines via branch entry.

Exit to:
For entry IGC0001D: Exit prolog.
For entry IEAVSPIE: Caller.
For entry IEAVSPI: Caller.
For entry IGC00040: Exit prolog.
For entry IGC00040+8: Caller.

IEAVTERM — Paging Termination Services

Operation: This module quiesces paging I/O for a particular RB or TCB. For a task being terminated, R/TM may also request that all pages fixed for a task be freed.

Entry from: At IEAVTERM from IEAVTSKT.

Exit to: IEAVTSKT.

IEAVTEST — TESTAUTH Service Routine

Operation: This routine tests the authorization of any program to perform a specified function.
For entry IGC119: Program authorization tester.
For entry IEAVTEST: Program authorization tester.

Entry from:
For entry IGC119: SVC IH.
For entry IEAVTEST: Supervisor routine via branch entry.

Exit to:
For entry IGC119: Exit prolog.
For entry IEAVTEST: Caller.

IEAVTMMT — Memory Termination Purges

Operation: This module handles the purging of resources held by a terminating memory.

Entry from: IEAVTRTE via BALR.

Exit-normal: Return to caller.

IEAVTMRM — R/TM Memory Resource Manager

Operation: This module frees all RTM2 work areas obtained from SQA for the terminating memory.

Entry from: IEAVTMMT via BALR.

Exit to: Caller.

IEAVTMSI — Recovery/Termination's Master Scheduler Initialization Controller

Operation: This module creates the following three recovery/termination tasks in the master memory:
- recording task
- memory termination control task
- SVC dump task

Entry from: Master scheduler routine IEEMB860 via link-macro to begin recovery/termination initialization function.

Exit-normal: To master scheduler IEEMB860 via PL/S return statement with return code 0 or 4.

Exit-error: Same as normal exit.

IEAVTMTC — Memory Termination Controller

Operation: This module controls the process of responding to a request for memory termination.

Entry from: IEAVTMSI via ATTACH.

Exit-normal: None. This is a never-ending task that waits on an ECB when there is no work to do.

IEAVTMTR — Memory Termination Requestor

Operation: This module is attached by the memory termination controller to invoke memory termination purges via SVC 13.

Entry from: IEAVTMTC via ATTACH.

Exit-normal: Return to caller.

IEAVTPMT — Type 1 Message Table Handler

Operation: This module prints or purges entries in the type 1 message table for the requested task.

Entry from:
IEAVTRTC via BALR for printing and purging.
IEAVTAS3 via BALR for purge.
IEAVTSKT via BALR for purging.

Exit to: Caller.

IEAVTRCE — Trace Routine

Operation: This module provides a trace function for the following events:

1) external interruptions,
2) I/O interruptions,
3) program interruptions,
4) SVC interruptions,
5) start I/O (from IOS),
6) task dispatcher exit conditions, and
7) event dispatcher exit conditions.

1. TREX: trace entry for external interruptions

Entry from: External IH.

Exit to: Caller via branch register 11.
2. TRIO: trace entry for I/O interruptions

Entry from: I/O Supervisor.

Exit to: Caller via branch register 11.
3. TRPI: trace entry for Program interruptions

Entry from: Program IH.

Exit to: Caller via branch register 11.
4. TRSVC: trace entry for SVC interruptions

Entry from: SVC IH

Exit to: Caller via branch register 11.
5. TRSIO: trace entry for start I/O

Entry from: Start I/O.

Exit to: Caller via branch.
6. TRDISP: trace entry for task dispatcher.

Entry from: Dispatcher.

Exit to: Caller via branch register 11.
7. TRSRB1: trace entry for event dispatcher dispatch

Entry from: Dispatcher.

Exit to: Caller via branch register 11.
8. TRSRB2: trace entry for event dispatcher redispatch

Entry from: Dispatcher.

Exit to: Caller via branch register 11.

IEAVTRER — Recording Request Routine

Operation: This module accepts requests either for recording on SYS1.LOGREC or for writing to the operator, and it schedules the recording/writing to occur asynchronously under the recording task. Also, if this is an emergency request from Machine Check Handler, the termination routine's emergency recorder returns the address and the length of scheduled but not yet written logrec request for recording.

Entry from: System routines via RECORD macro instruction.

Exit-normal: To caller.

Exit-error: To caller.

IEAVTRET — Asynchronous Recording Task

Operation: This module does the actual outputting of recording requests.

Entry from: IEVTMSI via ATTACH, subsequently by POST of RTCTRECB by IEVTRER via SRB.

Exit to: None. This is a never-ending task that waits on an ECB when there is no work to be done.

IEAVTRML — Installation Resource Manager List

Operation: This CSECT can be replaced with a CSECT containing a list of modules to be called by task and memory termination. It is defined to look like a list with no entries.

Entry from: not executable

Exits: None.

IEAVTRTC — RTM2 Mainline Controller

Operation: This module controls the processing among subfunctions, including task recovery, dumping, subtask handling.

Entry from: IEAVTRT2 via BALR.

Exit-normal: To IEAVTRT2.

IEAVTRTE — Control Task Purges, Memory Purges and Exit

Operation: This module invokes modules to perform task and memory purges and causes the appropriate exit from RTM2.

Entry from: IEAVTRT2 via BALR.

Exit-normal: To exit prologue or to STATUS to set task temporarily or permanently nondispatchable.

IEAVTRTH — RTM Hardware Error Processor

Operation: This module interfaces with hardware repair routines and it records a MCH LOGREC buffer to SYS1.LOGREC for all CALLRTM TYPE=MACKCK entries. The module will also accumulate data describing the error and its repair status for inclusion in the recovery routine interface.

Entry from: IEAVTRTM via BALR.

Exit to: Return to caller.

IEAVTRTM — Recovery Termination Manager Number 1

Operation: This module performs the following functions:

- interfacing with the module IEAVTRTH for recording and repairing damage after a hardware failure.
- interfacing with the System Recovery Manager module IEAVTRTS for routing control to the recovery routines which protect a supervisor path, and for recording software incidents.
- percolating errors to tasks or memories which are affected by the error.
- interfacing with the module IEAVTRTR which handles recovery for RTM1.
- handling the ABTERM, MEMTERM, and PGIOERR services.

Entry from: IEAVTRT1 via BALR.

Exit to: IEAVTRT1 via BALR.

IEAVTRTR — RTM Recovery Routines

Operation: This module contains the recovery routines for the RTM1 routines and also routines to perform Slip processing.

Entry from:

For entry FREEDCEL: This entry interfaces with the QUICKCELL routine to free any quickcells that are no longer needed.
For entry RTMRSFRR: This entry provides recovery for IEAVTRTM if a recursive error occurred during an attempt by R/TM to perform its reschedule function.
For entry RTHFRR: This entry provides recovery for IEAVTRTH.
For entry RTMSMFRR: This entry continues the recovery initiated by RTMRSFRR
For entry RCOVGETM: This entry defines a FRR that provides recovery for IEAVTRTS when a GETMAIN for a SDWA fails while processing an error in unlocked SRB mode.
For entry RCOVRCRD: This entry defines a FRR that provides recovery of IEAVTRTS in the event that recording an SDWA on behalf of a FRR fails.
For entry RCOVRGTF: This entry defines a FRR which provides recovery of IEAVTRTS in the event that GTF (invoked by IEAVTRTS to trace FRR recovery) should fail.
For entry RECVRRTM: This entry provides recovery for RTM1 during that part of its processing in which FRR protection is not possible or practical.
For entry SLIP: This entry determines if additional serviceability for the error which RTM1 is handling has been requested and, if so is it possible.
For entry RCOVSLP1: This entry defines a FRR which provides recovery for IEAVTRTS in the event that SLIP (invoked by IEAVTRTS) should fail.
For entry IEAVTRTL: This entry is used by SLIP entry and by RTM when a SLIP function is to be performed. The function if this entry is to schedule an SVC DUMP or to place the system in a wait state via module IEESTPRS.

For entry SLIP2FRR: This is the recovery code for the IEAVTRTL ENTRY function. It frees any acquired resources obtained by IEAVTRTL
Entry from: IEAVTRTS via LPSW or BALR; for FREEDCEL, IEAVTRTR or IEAVTRTM via BALR; for PECVRRTM, IEAVTRTM via BALR.

Exit to:
For entry FREEDCEL: Caller.
For entry RCOVRCRD: IEAVTRTS; or indirectly mate to IEAVTRTR entry point RECURRTM to terminate RTM1's processing of the error for error.
For entry RCOVRGTF: IEAVTRTS; or indirectly mate to IEAVTRTR entry point RECURRTM to terminate RTM1's processing of the error for error.
For entry RECVRRTM: IEAVTRTM or IEAVTRTS when RTM1's processing can continue; or to dispatcher IEAV0DS or SRB dispatcher when RTM1's processing of an error is terminated.
For entry RCOVSLP1: IEAVTRTS; or indirectly mate to IEAVTRTR entry point PECURRTM to terminate RTM1's processing of the error for error.
For all remaining entries: IEAVTRTS.

IEAVTRTS — System Recovery Manager

Operation: This module provides the interface and control between the control program and the functional recovery routines defined to recovery the control program.

- interfaces with the Software Recording facility (IEAVTRER) to record the software errors.
- interfaces with GTF to provide tracing of FRR recovery.
- interfaces with SLIP to activate any additional serviceability for the error being handled by RTM1.

Entry from: IEAVTRTM via BALR.

Exit-normal and error: Return to IEAVTRTM.

IEAVTRT1 — RTM1 Entry Point Processor

Operation: This module serves as an extension to the CALLRTM macro instruction and creates a common interface for the mainline RTM1.

Entry: PROGCK from caller via BALR. This entry gathers data pertinent to a program interruption for mainline RTM1.

Exit: to IEAVTRTM via BALR

Entry: RESTART from caller via BALR. This entry gathers data pertinent to a restart interruption for mainline RTM1.

Exit: to IEAVTRTM vai BALR

Entry: SVCERR from caller via BALR. This entry gathers data pertinent to a program issuing an SVC in a locked or SRB environment.

Exit: to IEAVTRTM via BALR

Entry: SVCERR from caller via BALR. This entry saves the caller's registers and establishes a recovery environment.

Exit: to IEAVTRTM via BALR

Entry: MACHCK from caller via BALR. This entry saves the caller's registers and establishes a recovery environment.

Exit: to IEAVTRTM via BALR

Entry: DATERR from caller via BALR. This entry gathers data pertinent to a translation failure for mainline RTM1.

Exit: to IEAVTRTM via BALR

Entry: ZABTERM from caller via BALR. This entry saves the caller's registers in a caller-provided save area and establishes a recovery environment.

Exit: to IEAVTRTM vai BALR

Entry: CABTERM from caller via BALR. This entry saves the caller's registers in an RTM WSA and establishes a recovery environment.

Exit: to IEAVTRTM via BALR

Entry: MEMTERM from caller via BALR. This entry saves the caller's registers in a caller-provided save are and establishes a recovery environment.

Exit: to IEAVTRTM via BALR

Entry: IEAVTRTN via LPSW. This entry is the machine check reentry, which sets up an interface for software recovery from a hardware error.

Exit: to IEAVTRTM via BALR

Entry: IEAVTRTX from SRB dispatcher via LPSW.

This entry is the cross address space ABTERM reentry, which sets up an interface for the ABTERM function of RTM1.

Exit: to IEAVTRTM via BALR

Entry: IEAVTRTZ for IEAVTRTM on return (BR 14). This entry sets up the final exit linkage back to the mainline supervisor as directed by RTM1.

Exit: exits are taken as follows:
Retry exit: register 15 is loaded with the retry address returned by RTM1.
MCH exit: BR 14.
Restart resume: LPSW of PSW stored by the restart interrupt.
Dispatcher exit: branch to address in CVT0DS.
SRB exit: branch to address in CVTSRBRT.
SVC exit: branch to address in CVTEXPRO.
to caller: BR 14.

IEAVTRT2 — RTM2 Initialization (SVC 13 Entry Point)

Operation: This module obtains and initializes the work area, and controls routing among major parts of RTM2 (including mainline, exit processing, and critical error handling).

Entry from: SVC FLIH.

Exit-normal: To IEAVTRTE via BALR.

Exit-error: Branch to dispatcher via STATUS.

IEAVTRV — Translate Real to Virtual

Operation: This module is given a real address as input. It returns the corresponding virtual address and ASID provided that the real storage page is assigned to a page of virtual space.

Entry from: At IEAVTRV from any routine.

Exit to: Caller.

IEAVTSBP — Task Recovery Resource Manager

Operation: During XCTL processing the ownership of eligible SCBs is transferred to the new RB resultant from the XCTL. In the case of an RB issuing exit or entry to TRRM for cleanup after an attach failure, the appropriate SCBs are purged from the Queue.

Entry from: IEAVEATO (Attach) via BALR to TRRM entry point address contained in CVTSRBP. from IEAVEDR (Exit) via BALR to TRRM entry point address contained in CVTSCBP. from (XCTL) via BALR to TRRM entry point address contained in CVTSRBP.

Exit-normal and error: To caller - with return code 0 or 4.

IEAVTSDI — SVC DUMP Initialization Routine

Operation: It acquires storage for a SDUMP work area and the SDUMP BUFFER. It also initializes the DUMP Data Set Table in the R/TM control table with with SYS1.DUMP Data Sets. It also places the entry point of IEAVTSDR Module in the CVT.

Entry from: IEAVNPA6 via BALR.

Exit-normal: Back to caller.

IEAVTSDR — SVC DUMP Resources Manager

Operation: This module will cleanup SVC DUMP resources if an address space fails in which SVC DUMP is active.

Entry from: IEAVTMMT via BALR.

Exit-normal: To IEAVTMMT.

Exit-error: To IEAVTMMT.

IEAVTSDT — SVC DUMP Task (one in every address space) attached at address space creation time

Operation: This module will issue an SVC 51 to invoke SVC DUMP in the address space that is the target of a scheduled SVC DUMP request.

Entry from: Attached at IEAVTSDT; awakened (via POST) at DUMPREQ

Exit normal: None. Never-ending task (as long as the address space exists).

Exit error: None. Never-ending task (as long as the address space exists)

IEAVTSDX — SVC DUMP Branch Entry Module

Operation: This module will schedule a SVC DUMP in the address space specified by the caller.

Entry from: IEAVTSDX via BALR 14, 15 from caller.

Exit-normal: Return to caller.

Exit-error: Return to caller.

IEAVTSIN — FRR Stack Initialization

Operation: IEAVTSIN is an executable macro included by IEAVNIPO and IEEVCPU to perform FRR stack initialization for a given CPU.

Entry from: This inline expansion is included by IEAVNIP0 (during system initialization) and by IEEVCPO (during vary CPU processing).

Exit-normal: To caller.

IEAVTSKT — Task Termination

Operation: This module handles the releasing of a terminating task resource.

Entry from: IEAVTRTE via BALR.

Exit-normal: To IEAVTRTE.

IEAVVCRA — Communication Task Console Attention Processor

Operation: The console attention processor receives control as a result of someone pressing the console's attention key. Pressing the attention key causes an attention interrupt. As a result of the attention interruption, this routine posts the communication task.

Exit to: A branch return to the POST status routine of the IOS handler based on the address in register 14.

IEAVVCRX — Communication Task Operator Interrupt Key Interrupt Processer

Operation: The operator interrupt key interrupt processor receives control from the external first level interrupt handler. This routine posts the communication task.

Entry from: IEEBC1PE.

Exit to: A branch return to the external first level interrupt handler based on the address in register 2.

IEAVVCTR — SVC 72 Router

Operation: Determines the appropriate device support processor (DSP) that will process the communications task service.

Entry from: Issuer of SVC 72.

Exit to:

- IEAV1052 for a 1052 printer-keyboard, 3210 console printer-keyboard, 3215 console pronter-keyboard, or 3213 console pronter supported as a console.
- IEAV1443 for a 1443 printer, 1403 printer, or 3211 printer supported as a console.
- IEAV2540 for a 2540 card reader punch, 2501 card reader, 2520 card reader punch, 3505 card reader, or 3525 card punch supported as a console.
- IEEC2740 for a 2740 communications terminal supported as a console.
- IEECVETW for a 3284/3286 printer supported as a console.
- IEECVETI for a graphics device supported as a console.

IEAVVRP1 — Reply Processor - Stage I

Operation: This module validates the command syntax and ID and schedules an SRB for stage II processing.

Entry from: IEE0403D.

Exit to: IEE0403D, IEE0503D.

IEAVVRP2 — Reply Processor - Stage II

Operation: This module checks the validity of a user's buffer area (for the text) and moves the reply text into the buffer if buffer is valid.

Entry from: IEAVVRP1 (via SRB).

Exit to: Caller.

IEAVVWTO — Communication Task Single-Line WTO and WTOR Service Routine (SVC 35)

Operation: The WTO and WTOR0 service routine processes the SVC 35 that is issued by the WTO and WTOR macro instruction expansion. The WTO and WTOR service routine also receives control for the multiple-line WTO (MLWTO) and write-to-programmer (WTP) (WTO) macro instructions; for these two, the WTO and WTOR service routine branches to the MLWTO (IEAVMWTO) and WTP (IGC0203E) routines respectively. When a WTO and WTOR message is to be sent to a console, the WTO and WTOR service routine posts the communication task.

Entry from: IEAVVWTO.

Exit to: A branch return to the SVC first level interrupt handler based on the address in register 14.

Error exit: When a WTO message has a length of zero bytes, the service routine does a branch return to the SVC first level interrupt handler based on the address in register 14.

Error exit: All other errors result in the user of WTO or WTOR macro instruction receiving a D23 ABEND.

IEAVXDOM — Communication Task DOM Service Routine (SVC 87)

Operation: The DOM service routine processes the SVC 87 that is issued by the DOM macro instruction expansion. This routine builds a DOM control block (DOMC) with the message identifications of the messages to be deleted and then posts the communication task.

Entry from: IGC0005G.

Exit to: A branch return to the SVC first level interrupt handler based on the address in register 14.

Error exit: The user of the DOM macro instruction will receive a 157 ABEND.

IEDAY3 — LOGON Synchronization Module

Operation: This routine synchronizes the LOGON process for TIOC by preventing memory creation from proceeding until the TIOC Logon routine has fully initialized the ASCB.

Entry from: IEEVWAIT.

Exit to: : IEEVEMCR.

IEEAB400 — WTP Buffer Routine

Operation: This module puts WTP messages into buffers and, if the messages are also WTO messages, issues the WTO macro instruction. When the buffer becomes full, this module calls IEEAB401 to issue the WTPs.

Entry from: IEFAB4FD.

Exit to: Return to caller.

Called Routines: IEEAB401.

IEEAB401 — WTP PUT Routine

Operation: This module issues the PUT macro instruction to write any WTP messages present in the buffer built by IEEAB400.

Entry from: IEEAB400, IEFAB4A0, IEFAB421, IEFBB401, IEFBB410, IEFAB4ED.

Exit to: Return to caller.

Called Routines: None.

IEECB800 — DISPLAY/TRACK Command Common Processor

Operation: This module invokes the proper routines to provide operand-based information to a console.

Entry from: IEEVWAIT.

Exit to: End of Task.

IEECB801 — Issuer of WTO and TPUT for DISPLAY and TRACK Commands

Operation: This module issues the multiline WTO (MLWTO) or iterative TPUTs for the DISPLAY and TRACK commands with operands JOBS, TS, and A.

Entry from: IEECB800.

Exit to: IEECB800.

IEECB860 — Command ESTAE Creation/Exit Routine

Operation: This module creates an ESTAE environment for the module that invokes it. If the ABEND STAE interface routine (ASIR) invokes the module, it provides a dump and messages.

Entry from: ABEND STAE interface routine.

Exit to: ABEND STAE interface routine.

IEECB866 — Console Dump

Operation: This module causes a dump of virtual storage to the current dump data set SYS1.DUMP.

Entry from: IEEVWAIT.

Exit to: System through use of SVC 3.

IEECB900 — VARY CN Syntax Processor

Operation: This module syntax checks the command and passes control to the authority processor for valid commands.

Entry from: IEEVWAIT.

Exit to: End of task.

Called Routines: IEECB901.

IEECB901 — VARY CN Console Processor

Operation: This module modifies a console authority value for eligible consoles.

Entry from: IEECB900.

Exit to: Return to caller.

IEECB904 — VARY Range Processor

Operation: This module creates a VARY command from a VARY device address-range command.

Entry from: IEE3603D, IEE3103D, IEE3303D, IEE4203D, IEE4603D.

Exit to: Caller.

IEECLEAN — Vary CPU/Channel Cleanup and Recovery Routine

Operation: For the Vary CPU/Channel function, this routine issues the messages and frees the system resources including storage for recorder blocks and SMF records.

Entry from: IEEVCPU.

Exit to: Caller.

IEECVETA — CONTROL Command Syntax Checker (DIDOCS)

Operation: Checks the syntax of the CONTROL command.

Entry from: IEECVET4.

Exit:

- To IEECVETK if roll mode or RTME (time interval specification) changes.
- To IEECVETD to issue warning message.
- To IEECVETH/P/R/U (device-dependent I/O modules) to rewrite the entry area.
- To IEECVETD to issue an error message.

IEECVETC — Asynchronous Error Processing Module (DIDOCS)

Operation: Initializes the DCM for a reopen; processes asynchronous errors; prepares for console switch after a permanent synchronous error.

Entry from:

- IEECVET1 for asynchronous error processing.
- IEECVFTG for asynchronous error processing.
- IEECVET4 for asynchronous error processing.

Exit:

- To IEECVETE to issue error message.
- To IEECVFTG to assist in error processing.
- To IEECVETK to set timer interval.
- To IEECVETH/P/R/U (device-dependent I/O modules) to write the screen image.
- To IEAVSWCH to perform console switch.

IEECVETD — Message Module (DIDOCS)

Operation: For full capability consodes, places error or informational messages in the warning line or the instruction line; for message stream or status display consoles, places messages in the warning line.

Entry from: Any DIDOCS module detecting a message-output condition.

Exit:
- To IEECVETH/P/R/U (device-dependent I/O modules) to write the screen.
- To IEECVET1 if hold mode is in effect.
- To IEECVET3 if roll mode is in effect and an unviewable message is needed.

IEECVETE — Message Module (DIDOCS)

Operation: Places messages into the instruction or warning line and places asynchronous-error messages into the entry area.

Entry from:
- IEECVETC to issue asynchronous-error message.
- IEECVET4 to issue error message.
- IEECVETK to issue error message.
- IEECVETF to issue error message.
- IEECVET6 to issue error message.
- IEECVET8 to issue error message.

Exit: To IEECVETH/P/R/U (device-dependent I/O modules) to write messages to the screen.

IEECVETF — Light-Pen and Cursor Detect Analyzer (DIDOCS)

Operation: Determines from the location of a light-pen or cursor detect what function is being requested.

Entry from: IEECVET1.

Exit:
- To IEECVET4 if detect occurred in out-of-line display.
- To IEECVETH/P/R/U (device dependent I/O modules) to preocess ENTER.
- To IEECVFTA to process PFK line detect.
- To IEECVET8 to process delete request.
- To IEECVET9 to process verification of a delete operation.
- To IEECVETD to issue error message.
- To IEECVETE to issue error message.
- To IEECVET1 to route ENTER for 3277 device.

IEECVETG — Open/Close Module (DIDOCS)

Operation: Performs open and close processing for cathode ray tube devices used as consoles.

Entry from:
- IEECVET1 if open pending.
- IEECVETK after it removes the console from roll mode.

Exit:
- To IEECLCTX if device was opened for console switch.
- To IEECVET1 after open to perform other communications task operations.
- To IEECVFTG to complete close of a console.
- To IEECVETK to remove console from roll mode before close.

IEECVETH — 3066 Device I/O Module (DIDOCS)

Operation: Performs input/output operations to 3066 devices.

Entry from: Any DIDIOCS module that modifies the SIB.

Exit:
- To IEECVET1 after I/O has been initiated.
- To IEECVFTG if status switch must take place.

IEECVETJ — Roll Mode Processor (DIDOCS)

Operation: Rolls messages off the console screen based on a timer interval and a number of lines to be rolled.

Entry from: IEECVET1.

Exit:
- To IEECVETK to display the number of messages waiting to be displayed.
- To IEECVET2 to handle timer supervision if no roll occurred and to display messages if space exists in the message area.

IEECVETK — Timer Interpreter (DIDOCS)

Operation: Resolves the different time intervals at which roll is to occur for different CRT consoles.

Entry from:

- IEECVET1 if timer expires.
- IEECVETC to reset time interval after asynchronous error.
- IEECVETG to remove a console from roll mode before closing the console.
- IEECVETA if roll mode or RTME (time interval) changes occur.

Exit:

- To IEECVET2 to display message after roll.
- To IEECVETE to display error message.
- To IEECVETD to display warning message.
- To IEECVETG to close the device.
- To IEECVETH/P/R/U to write the screen image on the console screen.
- To IEECVET1 if no roll is to occur.

IEECVETP — 2250 Device I/O Module (DIDOCS)

Operation: Performs I/O operations to 2250 devices.

Entry from: Any DIDOCS module that modifies the SIB.

Exit:

- To IEECVET1 after I/O has been initiated.
- To IEECVETF if light-pen or cursor detect occurs.
- To IEECVFTG for a status switch.

IEECVETR — 2260 Device I/O Module (DIDOCS)

Operation: Performs I/O operations to 2260 devices.

Entry from: Any DIDOCS module that modifies the SIB.

Exit:

- To IEECVET1 after I/O is initiated.
- To IEECVFTR for status switch.
- To IEECVET4 if cancel occurs.
- To IEECVETF if cursor detect occurs.

IEECVETU — 3277 Device and Model 158 System Console I/O Module (DIDOCS)

Operation: Performs I/O operations to 3277 devices and Model 158 system consoles.

Entry from: Any DIDOCS module that modifies the SIB.

Exit:

- To IEECVET1 after I/O is initiated.
- To IEECVETF if light-pen or cursor detect occurs outside entry area.
- To IEECVFTG for a status switch.
- To IEECVETK for roll mode.

IEECVETW — 3284/3286 Console Devcie Support Processor

Operation: Performs the following functions for 3284/3286 printers used as a console

-

- Opens the 3284/3286 as a console.
- Initiates a write operation to write messages to the console.
- After a write operation completes successfully, updates the console queue.
- When the 3284/3286 printer is to be removed from console status, closes the console.

Entry from: IEAVVCTR.

Exit:

- To IEAVSWCH if an I/O error occurs or the console cannot be opened.
- Otherwise, to issuer of SVC 72.

IEECVET1 — DIDOCS Router Module

Operation: Passes control to DIDOCS modules on the basis of requested function.

Entry from: IEAVVCTR.

Exit:

- To IEECVETG for console open and close.
- To IEECVETC for asynchronous error processing.
- To IEECVET4 for command processing.
- To IEECVET2 for message output.
- To IEECVET7 for message deletion.
- To IEECVET9 for message deletion.
- To IEECVFTP for erasing a status display or stopping a dynamic display.
- To IEECVETJ for roll.
- To IEECVETK for timer processing.
- To IEECVFTM for out-of-line message output.
- To IEECVETF for light-pen or cursor detect.
- To IEECVETH/P/R/U (device dependent I/O modules) to perform I/O.

- To IEECVFTA for PFK attention.
- To IEECVFTL for in-line message output.
- To IEECVFT1 for PFK redefinition.
- To IEECVFTT for display area blanking.
- Return to caller to perform other communications task operations.
- To IEECVETD to issue error messages.

IEECVET2 — In-line Message Output Module (DIDOCS)

Operation: controls the output of in-line messages.

Entry from:
- IEECVFT2 to determine exit from message output modules after single-line message output.
- IEECVET1 to display a message.
- IEECVFTL to detetmine exit from message output modules after multiple-line message output.
- IEECVETJ to display a message after roll.
- IEECVETK to display a message after roll.
- IEECVET9 to display a message after automatic deletion occurs.

Exit:
- To IEECVETH/P/R/U (device-dependent I/O modules to write messages to the screen.
- To IEECVET1 if there were no messages to display.
- To IEECVETD to issue a warning message.
- To IEECVFT2 to move messages to the DCM.
- To IEECVET7 to delete INTERVENTION REQUIRED messages.
- To IEECVET9 for automatic deletion of messages.

IEECVET3 — Message Output Module (DIDOCS)

Operation: Issues STIMER for roll mode after roll has been initialized.

Entry from: IEECVET2 to issue STIMER for roll mode.

Exit:
- To IEECVET1 if no I/O is to be performed after STIMER.
- To IEECVETD "Unviewable message" warning line is requested.
- To IEECVETH/P/R/U (device-dependent I/O modules) to write the screen image on the console screen.

IEECVET4 — Command Analyzer (DIDOCS)

Operation: Analyzes command input and routes the command to other DIDOCS modules or to the system for processing.

Entry from:
- IEECVET1 to process command input.
- IEECVETF to process a cancel key or light-pen detect on a cancel function.

Exit:
- To IEECVETC to complete asynchronous error processing.
- To IEECVET8 to initialize message deletion.
- To IEECVET9 to process delete verification.
- To IEECVET1 after command processing is complete.
- To the appropriate CONTROL command processing module.
- To IEECVETH/P/R/U (device dependent I/O modules) to rewrite the screen.

IEECVET6 — Message Deletion Module (DIDOCS)

Operation: Deletes messages requested by the K E,F or K E,nn(,nn) command.

Entry from: IEECVET4.

Exit:
- To IEECVET9 to remove messages.
- To IEECVETD for message verification.
- To IEECVETE to issue error messages.

IEECVET7 — Message Deletion Module (DIDOCS)

Operation: Deletes INTERVENTION REQUIRED messages and messages requested by DOM.

Entry from:
- IEECVET1.
- IEECVET2.
- IEECVET3.
- IEECVFT2.

Exit:

- To IEECVETH/P/R/U (device-dependent I/O modules) to write the screen.
- To IEECVET9 to attempt automatic deletion.
- To IEECVET1 on NOP entry.

IEECVET8 — Message Deletion Module (DIDOCS)

Operation: Deletes messages requested by the K E,SEG command, a cursor detect, or a light-pen detect.

Entry from:

- IEECVETF to process a light-pen or cursor detect.
- IEECVET4 to process the K E,SEG command.

Exit:

- To IEECVET9 to remove messages from the screen.
- To IEECVETD to issue a verification message.
- To IEECVETE to issue NO DELETABLE MESSAGE message.

IEECVET9 — Message Deletion Module (DIDOCS)

Operation: Processes automatic deletion when the screen is full, numbers messages in response to the K D,N command, deletes messages in response to a K command.

Entry from:

- IEECVET4 to process the K D,N command.
- IEECVET6 to delete messages if no verification is necessary.
- IEECVET7 to perform automatic deletion.
- IEECVET2 to perform automatic deletion if screen is full.

Exit:

- To IEECVET2 after automatic deletion.
- To IEECVETD to issue error messages.
- To IEECVETH/P/R/U (device dependent I/O modules) to rewrite the screen.

IEECVFTA — PFK-Entered Command Processor (DIDOCS)

Operation: Processes automatic command entry that results from a depressed PFK or a light-pen detect on the PFK line.

Entry from:

- IEECVET1 for depressed PFK.
- IEECVETF for light-pen detect on the PFK line.

Exit:

- To IEECVET1 to process an entered command (nonconversational mode).
- To IEECVETH/P/R/U (device-dependent I/O modules) to write the command to the entry area of the screen (conversational mode).
- To IEECVFTD to issue error messages.

IEECVFTB — PFK Definition Processor (DIDOCS)

Operation: Processes a PFK definition or redefinition in response to the K N,PFK command; displays or erases the PFK line in response to the K E,PFK or K D,PFK command.

Entry from: IEECVET4.

Exit:

- To IEECVETH/P/R/U (device-dependent I/O modules) to write or erase the PFK line.
- To IEECVFT1 to write a PFK definition to SYS1.DCMLIB.
- To IEECVETD to write error messages.
- To IEECVFTD to write error messages.

IEECVFTD — Message Module (DIDOCS)

Operation: Prepares for the writing of informational or error messages.

Entry from: IEECVFTA and IEECVFTB.

Exit: To IEECVETH/P/R/U (device-dependent I/O modules) to write messages to the screen.

IEECVFTG — DIDOCS Cleanup Module

Operation: Finishes functions that involve a system interface.

- If device is being closed, flags messages no longer needed.
- If device is changing status to message stream or full capability, flags out-of-line messages on the console queue.
- If device is changing status to status display, flags messages on console queue.
- If device is recovering from asynchronous error, flags out-of-line messages on the console queue.
- If this routine flags any out-of-line messages except for an asynchronous error, it stops a dynamic display, frees SACBs obtained with a GETMAIN, and resets the area definitions to their SYSGEN values.

Entry from:

- IEECVETG for open or close of a device.
- IEECVETC for an asynchronous error.
- IEECVETH/P/R/U (device-dependent I/O modules).

Exit:

- To IEECVET1 after close.
- To IEECVETC after all processing other than close.

IEECVFTL — In-line Multiple-line Message Processor (DIDOCS)

Operation: Controls the output of in-line, multiple-line messages (MLWTO).

Entry from:

- IEECVET1.
- IEECVFT2.

Exit:

- To IEECVFT2 to process next WQE.
- To IEECVET2 if roll mode is in effect and the screen is full.
- To IEECVETH/P/R/U (device-dependent I/O modules) to write the screen.
- To IEECVET1 if IEECVFTL encounters the end of the MLWTO chain but not the end of the message.
- To IEECVETD to issue error messages.

IEECVFTM — Out-of-line Multiple-line Message Processor (DIDOCS)

Operation: Controls the output of out-of-line, multiple-line messages (MLWTO).

Entry from: IEECVET1.

Exit:

- To IEECVFTO for display lines that overlay messages.
- To IEECVFTQ for display lines that do not overlay messages.
- To IEECVET1 if no more out-of-line messages are on the queue.
- To IEECVFTP for blanking of messages below a display.

IEECVFTN — Status Display Processor (DIDOCS)

Operation: Rewrites the display control line in response to a K D,F (framing), K D,H (holding), or K D,U (updating) command.

Entry from:

- IEECVET4 if console is a full-capability console.
- IEECVET1 if console is a status-display console.

Exit: To IEECVETH/P/R/U (device-dependent I/O modules) to write the screen.

IEECVFTO — Status Display Processor (DIDOCS)

Operation: Controls the output of a status display that overlays messages.

Entry from: IEECVFTM.

Exit: To IEECVETH/P/R/U (device-dependent I/O modules) to write three lines of the display area.

IEECVFTP — Status Display Processor (DIDOCS)

Operation: Controls the erasing of out-of-line displays.

Entry from: IEECVET1.

Exit:
- To IEECVETH/P/R/U (device-dependent I/O modules) to write the console screen.
- To IEECVFTT to blank the screen below the erased display.
- To IEECVET1 for a NOP entry.

IEECVFTQ — Status Display Processor (DIDOCS)

Operation: Controls the output of a status display to an out-of-line area that does not overlay messages.

Entry from: IEECVFTM.

Exit:
- To IEECVETH/P/R/U (device-dependent I/O modules) to write display lines to the screen.
- To IEECVFTM if no I/O is done.

IEECVFTT — Status Display Processor (DIDOCS)

Operation: Overlays with blanks any in-line message that is below an out-of-line display or between two out-of-line displays.

Entry from:
- IEECVET1.
- IEECVFTM.

Exit: To IEECVETH/P/R/U (device-dependent I/O modules) to write the screen.

IEECVFT1 — PFK Definition Processor (DIDOCS)

Operation: Updates PFK definitions in SYS1.DCMLIB.

Entry from: IEECVFTB.

Exit:
- To IEECVET1 after I/O to SYS1.DCMLIB has been initiated.
- To IEECVETH/P/R/U (device-dependent I/O modules) to blank the entry area.

IEECVFT2 — Single-line Message Processor (DIDOCS)

Operation: Displays and marks both action and deletable messages; locates and routes nondisplayed multiple-line messages (MLWTOs).

Entry from:
- IEECVET2 to move messages to the DCM.
- IEECVFTL to process the next WQE.

Exit:
- To IEECVET2 to continue output queue scan.
- To IEECVFTL to process in-line, multiple-line messages.

IEEC2740 — 2740 Console Device Support Processor

Operation: Performs the following functions for a 2740 communications terminal used as a console.
- Opens the 2740 communications terminal as a console.
- Uses BTAM to read a command from the console.
- After a read operation completes successfully, passes the command to command processing (SVC 34).
- Uses BTAM to write messages to the console.
- When the 2740 communications terminal is to be removed from console status, closes the console.

Entry from: IEAVVCTR.

Exit:
- To IEAVSWCH if an I/O error occurs or the console cannot be opened.
- Otherwise, to issuer of (SVC 72).

IEEDISPD — Display Domain Processor

Operation: Writes a console display of entries in the Domain Descriptor Table.

Entry from: IEEVWAIT.

Exit to: End of Task.

IEEJB840 — See IGC0203E

IEFJSWT — STC Write JCL Routine

Operation: This module writes the internal JCL text for a START, MOUNT, or LOGON command task into an appropriate system data set.

Entry from: IEESB605 via BALR.

Exit to: Caller via branch.

IEEMB803 — System Log Initialization/Writer Module

Operation: This module initializes the system log, opens and closes the log data set, and writes the log buffers to the log data set.

Entry from: IEEVWAIT.

Exit to: None - always available through waiting on ECB.

IEEMB804 — Write-To-Log (WTL) Processor — SVC 36

Operation: This module processes all valid WTL requests.

Entry from: IEE1603D.

Exit to: IEE1603D.

IEEMB805 — System Log Resource Initialization

Operation: This module acquires and initializes log resources used through log activation.

Entry from: IEEMB803.

Exit to: IEEMB803.

IEEMB806 — System Log ESTAE Processor

Operation: This module sets up two ESTAE environments for the system log, and it processes system log task ABEND situations. (The second ESTAE handles abnormal terminations of the first ESTAE).

Entry from: IEEMB803 (ESTAE/ABEND Interface).

Exit to: ESTAE/ABEND Interface.

IEEMB807 — System Log Message Module

Operation: This module issues messages for the system log task.

Entry from: IEEMB803, IEEMB806.

Exit to: Caller.

IEEMB810 — Performance Group Reset Module

Operation: This module passes performance group reset parameters to the system resources manager (SRM).

Entry from: IEEVWAIT.

Exit to: End of Task.

IEEMB811 — IPS Keywork Scanner

Operation: This module passes the (new) IPS keyword value to the system resources manager (SRM).

Entry from: IEEVWAIT.

Exit to: End of Task.

IEEMB812 — Set IPS Processor

Operation: This module invokes the IPS list processor to process the data in the IEAIPSxx parmlib member. It also notifies the system resources manager of IPS changes.

Entry from: IEEMB811.

Exit to: IEEMB811.

IEEMB813 — UNLOAD Command Syntax Scanner

Operation: This module scans the information in the UNLOAD command and validates the syntax.

Entry from: IEEVWAIT.

Exit to: End of Task.

IEEMB814 — IPS Scanner Message Module

Operation: This module contains messages for the IPS scanner routine.

IEESB670 — Job Scheduling Subroutine Recovery Exit Routine

Operation: This module schedules a retry of IEESB605 when one of the following conditions has occurred:

- A program check.
- A machine check.
- An ABEND.
- Depression of the restart key on a console.

If percolation from a lower level recovery routine has occurred, IEESB665 continues it.

Entry from: Recovery termination management via LINK.

Exit to: Caller via branch.

IEESTPRS — Stop/Restart Subroutine

Operation: This module stops the operating system in a manner that allows a restart interruption to bring the system back into operation. This module runs on one CPU and stops all others.

Entry from: IEEMPS03 and others.

Exit to: Caller.

IEEVALST — Validate Page Routine

Operation: This module sets to zero the storage and storage key of a 4K section of real storage.

Entry from: IEEMPVST.

Exit to: Caller.

IEEVCPU — Vary CPU/Channel Processor

Operation: This routine places a specified CPU or channel either online or offline.

Entry from: IEEVWAIT.

Exit to: System.

IEEVDEV — Device Subroutine

Operation: This routine obtains information regarding the condition of available paths to a device and provides that information to the caller of the routine.

Entry from: Allocation; DDR; VAP; VARY CPU; VARY PATH.

Exit to: Caller.

IEEVIPL — Master Scheduler Base Initialization

Operation: This module initializes the scheduler, invokes the subsystem interface initialization function and other initializers.

Entry from: NIP.

Exit to: None — it is a non-ending task.

IEEVJCL — Job Control Language Build Routine

Operation: This module builds internal JCL text for a command task that represents a START, MOUNT, or LOGON command.

Entry from: IEEVSTAR via branch, for a START command; from IEEVMNT1 via branch, for a MOUNT command; from IKJEFLA via branch, for LOGON commands.

Exit to: IEESB605 via XCTL.

IEEVMNT1 — MOUNT Command Syntax Check Routine

Operation: This module checks a MOUNT command and its parameters for correct syntax; it also uses this input to build a START descriptor table (SDT) that contains internal JCL.

Entry from: IEEPRWI2 via XCTL.

Exit to: IEEVJCL via branch for normal processing; to IEEPRTN2 via XCTL for error processing.

IEEVMNT2 — MOUNT Command Processor

Operation: This module checks the unit control block (UCB) associated with the device to be mounted. If the device is marked permanently resident or reserved, IEEVMNT2 issues an error message. Otherwise, it also checks the MOUNT command input buffer (CIB) for the use attribute specified for the device.

Entry from: IEFSD263, an initiator routine, via ATTACH.

Exit to: IEFSD263 via branch.

IEEVMSG — STC Message Module

Operation: This module contains the text of messages issued by the started task control routines.

Entry from: IEEVMNT2 or IEESB605 via CALL.

Exit to: Caller via branch.

IEEVPTH — VARY Path Command Processor

Operation: This routine makes individual paths to a device either available or unavailable for I/O processing.

Entry from: IEEVWAIT.

Exit to: System.

IEEVSEND — Operator SEND Command Main Control

Operation: This module provides a console or TSO-terminal operator with a means of communicating with other TSO and/or console operators.

Entry from: IEEVWAIT.

Exit to: End of Task.

IEEVSND2 — SEND Command Mail Handler

Operation: This module saves the mail for specified users in the mail section of the SYS1.BRODCAST data set.

Entry from: IEEVSND9.

Exit to: IEEVSND9.

IEEVSND3 — SEND Command List Handler

Operation: This module lists all notices from or recovers a specific notice from, the SYS1.BRODCAST data set for the sending operator.

Entry from: IEEVSND9.

Exit to: IEEVSND9.

IEEVSND4 — SEND Command Cleanup Routine

Operation: This module issues informational, warning, and error messages to the operator.

Entry from: IEEVSEND.

Exit to: IEEVSEND.

IEEVSND5 — Operator SEND Command I/O Routine

Operation: This module performs I/O operations for the SEND command.

Entry from: IEEVSND2; IEEVSND3; IEEVSND8, IEEVSND9.

Exit to: Caller.

IEEVSND6 — SEND Command 'Now' Processor

Operation: This module issues the SEND command text to the specified recipients.

Entry from: IEEVSEND.

Exit to: IEEVSEND.

IEEVSND8 — SEND Command Notice Handler

Operation: This module adds notices to or deletes notices from the SYS1.BRODCAST data set.

Entry from: IEEVSND9.

Exit to: IEEVSND9.

IEEVSND9 — SYS1.BRODCAST Data Set Access Controller

Operation: This routine coordinates the operator usage of the SYS1.BRODCAST data set.

Entry from: IEEVSEND.

Exit to: IEEVSEND.

IEEVSTAR — START Command Syntax Check Routine

Operation: This module checks a START command and its parameters for correct syntax; it uses this input to build a START descriptor table (SDT) that contains the internal JCL for the task to be started.

Entry from: IEEPRWI2 via XCTL.

Exit to: IEEVJCL via branch, for normal processing; to IEEPRTN2 via XCTL, for error processing.

IEEVSTOP — Vary CPU Stop Routine

Operation: This routine sets the prefix register of the executing CPU to zero and issues a SIGP STOP instruction to stop that CPU (that is, the CPU on which the routine is executing).

Entry from: IEEVCPU, Machine Check Handler, Timer.

Exit to: None.

IEEVWAIT — Master Scheduler Wait

Operation: IEEVWAIT, at system initialization time, terminates system trace (if necessary), and attaches the system log task. After that, IEEVWAIT's only function is to scan the CSCB chain and process any pending commands.

Entry from: IEEMB860 at system initialization time; IEE0803D (via POST).

Exit: No normal exit. This is a permanent task.

Error entry: At entry point STAE0000 from failure or from ABEND.

Error exit: To a wait state if restart fails.

IEEVWKUP — Vary CPU Wakeup and Quiet Routine(s)

Operation: This routine performs one of three functions:

- It completes the preliminary initialization for the CPU that is being brought online.
- It performs a quiesce function for the vary offline process. This includes resetting control registers to their initial values.
- It initializes the channel availability table in the physical configuration communication area of the executing CPU.

Entry from: IEEVCPU.

Exit to: Caller.

IEEXEDNA — Display Consoles Command Processor

Operation: This module writes a console status display to the console that issued the command. For input stream commands, the display goes to the master console. If L=cca is specified, output goes to the routed console, which may be different from the one issuing the command.

Entry from: IEEVWAIT.

Exit to: Dispatcher or End of Task.

IEE0003D — ESTAE Environment Creation Routine

Operation: This routine creates an ESTAE environment routine to protect the command scheduler from ABEND situations.

Entry from: Issuer of SVC 34.

Exit to: IEE0303D.

IEE00110 — Master Scheduler SVC 110 Router

Operation: This module routes control for the processing of the commands D C,K; D U; and D PFK.

Entry from: IEEPALTR.

Exit to: IEE10110, IEE20110, IEE40110.

IEE0303D — Control Block Chain Manipulator; QEDIT and MGCR Macro Function Processor

Operation: This module manipulates CSCB and CIB chains. It also communicates ABTERM requests to the ABTERM routines, and it is the QEDIT processor.

Entry from: IEE0003D.

Exit to: IEE5403D.

IEE0403D — Command Scheduler Router

Operation: This module scans, identifies, and verifies commands. It gives control to the proper command processor.

Entry from: IEE5403D.

Exit to: IEE0503D; Command Processors.

IEE0503D - Command Processing Message Assembly

Operation: This module assembles and edits messages for command processors. It issues WTO or TPUT macro instructions to write the message.

Entry from: Many.

Exit to: Caller.

IEE0603D — Immediate Routine for SET Command

Operation: This module inspects the operands of the SET command and routes control to the appropriate processing module.

Entry from: IEE0403D.

Exit to: IEE6503D, IEE6603D, IEE0803D, IEE0503D.

IEE0703D — MODIFY and STOP Command Processor

Operation: This module performs processing for the STOP and MODIFY commands.

Entry from: IEE0403D.

Exit to: IEE0003D.

IEE0803D — CSCB and ASCB Creation

Operation: For Task-creating commands, this module builds a CSCB. For memory-creating commands, this module interfaces with memory-requesting routines.

Entry from: Variable.

Exit to: IEE0003D, (Posts IEEVWAIT).

IEE10110 — Display C, K Processor, Load 1

Operation: This module writes the first twelve lines of a status display showing the K command operands.

Entry from: IEE00110.

Exit to: IEE11110.

IEE11110 — Display C, K Processor, Load 2

Operation: This module writes lines 13 through 26 of a status display showing the operands of the K command.

Entry from: IEE10110.

Exit to: IEE12110.

IEE12110 — Display C, K Processor, Load 3

Operation: This module writes lines 27 through 38 and the last line of a status display showing the operands of the K command.

Entry from: IEE11110.

Exit to: Caller.

IEE1403D — HALT and SWITCH Command Syntax Checker

Operation: This module preprocesses the HALT and SWITCH commands by checking the command syntax.

Entry from: IEE0403D.

Exit to: IEE0803D, IED1303D, ISTCFF3D.

IEE1603D — LOG and WRITELOG Command Processor

Operation: This module processes the commands LOG and WRITELOG (with the options CLOSE, START, or CLASS).

Entry from: IEE0403D.

Exit to: IEE0003D.

IEE20110 — Unit Status, Load 1

Operation: This module analyzes the operands of a D U command, gets a work area, and finds the first UCB for the first unit to be displayed.

Entry from: IEE00110.

Exit to: IEE23110, IEE22110.

IEE21110 — Unit Status, Load 2

Operation: This module loads the device name table the first time it is needed, and it creates the text of each line to be displayed.

Entry from: IEE23110.

Exit to: IEE23110.

IEE22110 — Unit Status, Load 3

Operation: This module issues error messages, frees workarea storage, and deletes the device name table.

Entry from: IEE20110, IEE23110.

Exit to: Caller.

IEE2303D — SMF VARY Record Handler

Operation: This module writes an SMF VARY ONLINE record.

Entry from: IEE3303D.

Exit to: IEE4203D; IEE4403D.

IEE23110 - Unit Status, Load 4

Operation: This module determines the next unit address to be displayed and issues the WTO for the lines of the display.

Entry from: IEE20110, IEE21110.

Exit to: IEE21110, IEE22110.

IEE2903D — Display Requests Routine

Operation: This module causes the user's console to display the ID of each WTOR type message to which a reply has not been given. Mount information and operator intervention information is also displayed.

Entry from: IEE7503D.

Exit to: IEE0003D.

IEE3103D — VARY ONLINE/OFFLINE Processor

Operation: This module varies non-console units to either an online or an offline status.

Entry from: IEE4203D.

Exit to: End of Task, IEECB904.

IEE3203D — VARY Keyword Scan

Operation: This module scans first the primary keywords then the secondary keywords of the VARY command. It passes control according to the keyword found.

Entry from: IEE0403D.

Exit to: Variable.

IEE3303D — VARY ONLINE/OFFLINE/CONSOLE Syntax Scan

Operation: This module checks the unit field syntax, command authority, and presence of SMF in order to route control to the appropriate processor.

Entry from: IEE3603D.

Exit to: IEE4403D, IEE4203D, IEE2303D, IEECB904.

IEE3503D — DISPLAY/TRACK Router

Operation: This module routes control to a processor module based on the command operand. It processes the D T command itself.

Entry from: IEE0403D.

Exit to: Variable.

IEE3603D — MP VARY Command Pre-processor

Operation: This module presets the environment for the VARY ONLINE/OFFLINE/CONSOLE commands. It also keeps offline all devices without available paths.

Entry from: IEEVWAIT, IEECB904.

Exit to: IEE3303D, IEECB904.

IEE3703D — CANCEL Command Handler

Operation: This module cancels any background and foreground (TSO) jobs that are currently executing.

Entry from: IEE0403D.

Exit to: IEE0003D.

IEE40110 — Display PFK Processor

Operation: This module locates and displays the program function key definitions currently in effect for the operator's display console.

Entry from: IEE00110.

Exit to: Caller.

IEE4103D — Hardcopy Informational Message Module

Operation: This module issues a hardcopy information message for devices that have been varied as a hardcopy device.

Entry from: IEE7203D.

Exit to: IEE0003D.

IEE4203D — UCME Scanner for VARY Command

Operation: This module does syntax validation for VARY ONLINE/OFFLINE/CONSOLE commands.

Entry from: IEE4403D, IEE3303D, IEE2303D.

Exit to: IEE4903D, IEE3103D, IEE4603D, IEECB904.

IEE4303D — Processor for VARY Master Console Command

Operation: This module processes the VARY MSTCONS command for the console switch routine.

Entry from: IEE3203D.

Exit to: IEE0003D.

IEE4403D — VARY Keyword Scanner

Operation: This module scans the input buffer and validates keyword values.

Entry from: IEE3303D, IEE2303D.

Exit to: IEE4203D.

IEE4603D — VARY ONLINE/OFFLINE for Console Devices

Operation: This module processes the VARY command keywords ONLINE and OFFLINE.

Entry from: IEE4203D.

Exit to: IEE3103D, IEECB904, SVC 3 (IGC003).

IEE4703D — VARY HARDCPY Operand Processor

Operation: This module begins the processing of a VARY HARDCPY Command.

Entry from: IEE3203D.

Exit to: IEE5703D, IEE7203D.

IEE4803D — Console Information Message Routine

Operation: This module uses the UCM and UCB to construct a console status message.

Entry from: IEE4903D.

Exit to: IEE7303D.

IEE4903D — Console Operand Processor

Operation: This module processes the CONSOLE Operand for the VARY command.

Entry from: IEE4203D.

Exit to: IEE4803D.

IEE5103D — ESTAE Exit Routine

Operation: This module is the SVC 34 ESTAE exit routine. It provides a storage dump in response to an ABEND situation, and it checks the validity of entries in the CSCB and CIB chains.

Entry from: ABEND STAE interface routine.

Exit to: Termination retry address in parameter list

IEE5403D — Command Translation Routine

Operation: This module translates lower case letters, initializes the XSA, finds the beginning of a command verb and writes the command to hard copy if required.

Entry from: IEE0303D.

Exit to: IEE0403D.

IEE5503D — Stop Track and Stop Monitor Processor

Operation: This module processes the operands for both the STOPTR and STOPMN commands.

Entry from: IEE7503D, IEE0403D.

Exit to: IEE6703D, IEE0003D, IEE7103D, IEE0503D.

IEE5603D — MSGRT and CONTROL Command Message Module

Operation: This module issues error messages for CONTROL and MSGRT commands and for L=cca operands.

Entry from: Variable.

Exit to: IEE5903D, IEE0503D, or to system.

IEE5703D — VARY HARDCPY, OFF Processor

Operation: This module removes the hardcopy function from a system when requested.

Entry from: IEE4703D.

Exit to: IEE0003D.

IEE5903D — Error Message Module 2 for CONTROL and MSGRT Commands

Operation: This module handles the second parts of messages for the appropriate commands and also for the L=cca operand errors.

Entry from: IEE5603D.

Exit to: IEE0003D.

IEE6303D — MSGRT Command Handler I

Operation: This module builds the route control table (RCT) for each console. This table provides default routing values.

Entry from: IEE0403D.

Exit to: IEE0503D, IEE6403D.

IEE6403D — MSGRT Command Handler II

Operation: This module formats a MSGRT command on the basis of values in the route control table.

Entry from: IEE6303D.

Exit to: IEE0003D, IEE5603D.

IEE6503D — Set Local Time Routine

Operation: This module changes or resets the local time and date. It also updates the real time timer queue element (TQE) queue.

Entry from: IEE0603D.

Exit to: IEE0603D.

IEE6603D — Set Time-of-Day Clock Routine

Operation: This module sets a TOD clock with a time and date value that has been specified by an operator.

Entry from: IEE0603D.

Exit to: Caller.

IEE6703D — Processor for CONTROL and STOPTR Commands

Operation: This module handles the syntax checking of the K command and schedules the STOPTR command and the K command.

Entry from: IEE7503D.

Exit to: Variable.

IEE6803D — K A,nn,nn Command Handler

Operation: This module handles the processing of the K A command, which defines CRT screen areas.

Entry from: IEE6703D.

Exit to: IEE0003D, IEE5603D.

IEE6903D — K A and K T Command Handler

Operation: This module handles K T, K A, and K A,none commands.

Entry from: IEE6703D.

Exit to: IEE0003D, IEE5603D.

IEE70110 — Halt-End-of-Day and Switch-SMF Processor

Operation: This module switches SMF data sets and/or halts system processing.

Entry from: IEEVWAIT.

Exit to: End of Task.

IEE7103D — MONITOR Command Processor

Operation: This module processes the MONITOR command operands.

Entry from: IEE0403D

Exit to: IEE0003D, IEE0503D

IEE7203D — VARY HARDCPY Processor II

Operation: This module changes the hardcopy configuration in a multiple console support (MCS) environment.

Entry from: IEE4703D.

Exit to: IEE4103D.

IEE7303D — Console Information Message Routine 2

Operation: This module constructs a message that displays the current status of a console.

Entry from: IEE4803D.

Exit to: End of Task.

IEE7503 — Routing and List Operand Processor

Operation: This module processes the routing and/or LIST operands for DISPLAY, TRACK, CONTROL, and STOPTR commands.

Entry from: IEE3503D, IEE0403D, JES2 Routine.

Exit to: IEE2903D, IEE0803D, IEE5503D, JES2 Routine.

IEE7703D — CONTROL V Command Handler

Operation: This module processes CONTROL V commands.

Entry from: IEE6703D.

Exit to: IEE0003D, IEE5603D.

IEE7803D — CONTROL C,D Command Handler

Operation: This module processes the CONTROL C,D command for cancelling an inline MLWTO.

Entry from: IEE6703D.

Exit to: IEE0003D, IEE5603D.

IEE8603D — SETDMN Command Processor

Operation: This module passes new domain description values to the system resources manager (SRM).

Entry from: IEE0403D.

Exit to: IEE0003D.

IEE90110 — HALT/SWITCH Message Module

Operation: This module assembles and edits messages for HALT and SWITCH commands, and issues a WTO.

Entry from: IEE70110.

Exit to: IEE70110.

IEE9403D — MSS Preprocessor

Operation: This module prepares MSS commands for interfacing to MSS routines, if MSS is active.

Entry from: IEE0403D, IEE1403D, IEE3203D.

Exit to: Posts MSS routines and then returns to caller.

IEFAB4A0 — Common Unallocation Control

Operation: This module controls the processing necessary to unallocate a request that has been allocated by means of either JCL or a dynamic allocation. The functions provided are:

- disposition processing
- disconnecting of private catalogs
- data set release (DEQ)
- unit unallocation
- volume release (DEQ)
- release of DSAB/TIOT entries.

Entry from: IEFAB4DE, IEFAB477, IEFAB492, IEFBB414, IEFBB416, IEFDB4A1, IEFDB413, IEFDB418, IEFDB490.

Exit to: Return to caller.

Called Routines: IEEAB401, IEFAB4A2, IEFAB4A4, IEFAB4A6, IEFAB4A8, IEFAB4FC, IEFAB4F4.

IEFAB4A2 — Disposition Processing Control

Operation: This module controls the processing required to dispose of data sets as indicated in the input requests.

Entry from: IEFAB4A0.

Exit to: Return to caller.

Called Routines: IEFAB4A3, IEFAB4B0, IEFAB4B2.

IEFAB4A3 — Special Scratch Processing

Operation: This module builds a dummy ETIOT needed by the scratch function of data management, when no DSABs and TIOTs exist.

Entry from: IEFAB4A2.

Exit to: Return to caller.

Called Routines: IEFAB4FC, IEFAB428.

IEFAB4A4 — Unit Unallocation Control

Operation: This module controls the processing associated with unallocating devices for each request. The functions performed are:

- Releasing units no longer needed.
- Rewinding tape volumes.
- Deleting outstanding mount messages.
- Identifying volumes to be unloaded.
- Posting waiting allocations.

Entry from: IEFAB4A0, IEFAB4DE.

Exit to: Return to caller.

Called Routines: IEFAB4FA, IEFAB4F7, IEFAB49C.

IEFAB4A6 — Data Set Release

Operation: This module releases (dequeues) data sets that are no longer needed by the job.

Entry from: IEFAB4A0, IEFBB401.

Exit to: Return to caller.

Called Routines: IEFAB4F7.

IEFAB4A8 — Volume Release

Operation: This module is responsible for releasing volumes that were reserved for the job step by allocation and data management (OPEN/CLOSE/EOV).

Entry from: IEFAB4A0.

Exit to: Return to caller.

Called Routines: None.

IEFAB4B0 — Disposition Message Routine

Operation: This module issues disposition messages to the system output data set and to the system operator.

Entry from: IEFAB4A2.

Exit to: Return to caller.

Called Routines: IEFAB4FD.

IEFAB4B2 — Alternate Disposition Message Routine

Operation: This module issues disposition messages to the system output data set and/or to the system operator when no storage was available for a volume list.

Entry from: IEFAB4A2.

Exit to: Return to caller.

Called Routines: IEFAB4FD.

IEFAB4DC — Data Set Reservation/Release

Operation: This module can perform the following functions:

- Enqueue a dsname for shared or exclusive use.
- Update the data set enqueue table to add or change an entry.
- Dequeue the dsname.
- Update the data set enqueue table to delete an entry.

Entry from: IEFAB459, IEFDB411, IEFDB413, IEFDB418.

Exit to: Return to caller.

Called Routines: IEFAB4F7.

IEFAB4DD — Job/Step Unallocation ESTAE Exit

Operation: In the event of an abnormal termination, this module calls step and/or job unallocation, if needed and if SWA records have been read successfully. This module also requests a dump, if necessary.

Extry from: R/TM (Recovery/Termination Management).

Exit to: Return to caller.

Called Routines: IEFBB414, IEFBB416.

IEFAB4DE — Common Unallocation ESTAE Exit Routine

Operation: This module attempts to perform any common unallocation functions that had been requested but not yet tried at the time of an abnormal termination. It also requests a dump, if necessary.

Entry from: R/TM (Recovery/Termination Management).

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB4A4, IEFAB4FA.

IEFAB4EA — Housekeeping ESTAE Exit Routine

Operation: This module reestablishes control block pointers and frees resources for JFCB Housekeeping Control (IEFAB451), Allocate Catalog Control (IEFAB4F5), or Unallocate Catalog Control (IEFAB4F4), if an ESTAE is scheduled in either of those routines as the result of an ABEND.

Entry from: IEFAB4ED.

Exit to: Return to caller.

Called Routines: IEFAB4F4.

IEFAB4EB — PDI Read and Chain

Operation: This module reads all the PDIBs (Passed Data set Information Blocks) that comprise the PDI (Passed Data Set Information) and chains the PDIB(s) together.

Entry from: IEFAB455, IEFAB459, IEFBB416.

Exit to: Return to caller.

Called Routines: IEFAB4F7.

IEFAB4EC— Group Lock/Unlock Interface

Operation: This module locks the groups represented by the list of UCBs passed to it. It performs this serialization via the Allocation Queue Manager.

Entry from: IEFBB416.

Exit to: Return to caller.

Called routines: IEFAB4FA and IEFAB4UV.

IEFAB4ED— Allocation Common ESTAE Exit

Operation: This module is the common ESTAE exit for all of allocation. It performs general recovery processing and then routes control to other exit routines for more specific recovery.

Entry from: R/TM (Recovery/Termination Management).

Exit to: Return to caller.

Called Routines: IEEAB401, IEFAB4DD, IEFAB4DE, IEFAB4EA, IEFAB4E2, IEFAB4E4, IEFAB4E7, IEFAB4E8, and IEFDB402.

IEFAB4EE — Allocation Message Routine

Operation: This module builds the allocation messages for the requests on the input SIOT chain.

Entry from: IEFAB490, IEFBB414.

Exit to: Return to caller.

Called Routines: IEFAB4FD, IEFAB4F3.

IEFAB4EF — PCCB Routine

Operation: This module handles all requests for the creation, modification, and deletion of PCCB(s) and their chain structure.

Entry from: IEFAB4F4, IEFAB4F5, IEFAB452.

Exit to: Return to caller.

Called Routines: None.

IEFAB4E0 — Test If Device Is Ready

Operation: This module tests to determine whether a device is physically ready by issuing a NOP EXCP to the device.

Entry from: IEFAB473, IEFAB49C, IEFAB494, IEFAB495.

Exit to: Return to caller.

Called Routines: None.

IEFAB4E1 — VM&V FRR Exit Routine

Operation: This module receives control if an ABEND occurred while an MVCA was being dechained. It tries again to take the MVCA off the chain. If successful, it sets a bit to indicate that it should free MVCA storage.

Entry from: R/TM (Recovery/Termination Management).

Exit to: Return to caller.

Called Routines: None.

IEFAB4E2 — VM&V ESTAE Exit Routine

Operation: In the event of an abnormal termination, this module releases mount control blocks that still exist for this allocation.

Entry from: IEFAB4ED.

Exit to: Return to caller.

Called Routines: IEFAB49A, IEFAB498.

IEFAB4E3 — Job/Step Allocation Retry Routine

Operation: This retry routine receives the initiator's registers on entry. It sets a return code of 4 and branches on register 14, thereby simulating an error return to the initiator.

Entry from: R/TM (Recovery/Termination Management).

Exit to: IEFSD162.

Called Routines: None.

IEFAB4E4 — Job/Step Allocation ESTAE Exit

Operation: This module sets bits in the LCT to indicate that an error has occurred. It then prepares for retry by storing the initiator's registers in the SDWA. These registers will be received by retry routine IEFAB4E3.

Entry from: IEFAB4ED.

Exit to: Return to caller.

Called Routines: None.

IEFAB4E5 — Allocation Resource Manager

Operation: This module attempts to clean up system resources owned by allocation in the event of an abnormal memory termination. It does the following:

- Releases device groups held by the terminating address space.
- Frees cross memory communication areas.
- Posts other allocations waiting for devices.
- Unallocates non-shareable UCBs allocated to this address space.

Entry from: R/TM (Recovery/Termination Management).

Exit to: Return to caller.

Called Routines: IEFAB4FA, IEFAB498.

IEFAB4E6 — Update UCB FRR Exit Routine

Operation: This module checks to ensure that the use count in the UCB has been updated if the UCB had been marked allocated prior to the abnormal termination.

Entry from: R/TM (Recovery/Termination Management).

Exit to: Return to caller.

Called Routines: None.

IEFAB4E7— Group Lock/Unlock ESTAE Exit

Operation: This module frees resources held by LOCK/UNLOCK processing (IEFAB4EC) if an ABEND occurs while IEFAB4ECs ESTAE is in effect.

Entry from: IEFAB4ED.

Exit to: Return to caller.

Called Routines: IEFAB4FA.

IEFAB4E8 — Common Allocation ESTAE Exit Routine

Operation: This module performs clean-up processing if an abnormal termination has occurred during common allocation processing.

Entry from: IEFAB4ED.

Exit to: Return to caller.

Called Routines: IEFAB4FA, IEFAB49A, IEFAB498.

IEFAB4E9 — Dynamic Allocation Set DSABDCBM Mask Bits Subroutine

Operation: For each filled-in DCB field in the JFCB, this module sets on the corresponding bit in DSABDCBM.

Entry from: IEFAB428, IEFDB411.

Exit to: Return to caller.

Called Routines: None.

IEFAB4FA — Allocation Queue Manager

Operation: This module serializes access to UCB (device) groups by device allocation and manages the WAIT-FOR-DEVICE queue.

Entry from: IEFAB4A4, IEFAB4DE, IEFAB4E5, IEFAB4E8, IEFAB471, IEFAB472, IEFAB485, IEFAB486, IEFAB490, IEFAB491, IEFAB492.

Exit to: Return to caller.

Called Routines: IEA0PT01 (System Post).

IEFAB4FC — TIOT Manager Control Routine

Operation: This module is responsible for obtaining and maintaining the TIOT and the DSAB queue descriptor block (QDB). The functions of this routine are:

- Create TIOT and DSAB QDB.
- Allocate a TIOT DD entry.
- Release a TIOT DD entry.
- Update TIOT and DSAB QDB.
- Unallocate DSAB/TIOT(ETIOT) entry.
- Concatenate DSAB/TIOT DD entry.
- Free TIOT and QDB.

Entry from: IEESB601, IEFAB4A0, IEFAB4A3, IEFAB428, IEFBB401, IEFBB404, IEFBB410, IEFDB413, IEFDB450.

Exit to: Return to caller.

Called Routines: None.

IEFAB4FD — System Message Interface Routine

Operation: This module converts the input parameter list into a WTO parameter list, which it passes to IEEAB400. IEEAB400 buffers any WTO messages (routing code 11) and issues the WTO macro instruction if any other routing codes are specified.

Entry from: IEFAB4B0, IEFAB4B2, IEFAB4EE, IEFAB490, IEFBB401, IEFBB402, IEFBB404, IEFBB410.

Exit to: Return to caller.

Called Routines: IEEAB400.

IEFAB4FE — SWA Reader Routine

Operation: This module establishes addressability to SIOTs and their related JFCBs and JFCBXs.

Entry from: IEFBB401, IEFBB410, IEFDB400.

Exit to: Return to caller.

Called Routines: IEFAB4F7.

IEFAB4F0 — Generalized Conditional ENQ/DEQ Routine

Operation: This module does the following:

- ENQs conditionally (either shared or exclusive) on the input volume serial number.
- DEQs conditionally on the input volume serial number.

Entry from: IEFAB436, IEFAB479, IEFAB492.

Exit to: Return to caller.

Called Routines: None.

IEFAB4F1 — Test Volume ENQ

Operation: This module determines if the input volume serial number can be shared by this allocation (even though it is already being used by this job step through a prior entry to allocation).

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: None.

IEFAB4F2 — Update Algorithm Tables

Operation: This module helps maintain the cover/reduce algorithm interface tables. Its functions are:

- Change a request to device-required.
- Indicate a request was just allocated.

Entry from: IEFAB432, IEFAB434, IEFAB442.

Exit to: Return to caller.

Called Routines: IEFAB481.

IEFAB4F3 — Message Compression Routine

Operation: This module eliminates consecutive and trailing blanks from the text received in the input buffer. The compressed text and the new text length are returned to the caller.

Entry from: IEFAB4EE, IEFAB421, IEFAB48A, IEFAB487, IEFAB488, IEFAB490, IEFBB401, IEFBB404, IEFBB410, IEFBB416.

Exit to: Return to caller.

Called Routines: None.

IEFAB4F4 — Catalog Unallocation Control

Operation: This module calls other routines to perform the following functions, as requested by the caller:

- close private catalogs
- unallocate private catalogs
- release private catalog control blocks (PCCBs).

Entry from: IEFAB4A0, IEFAB4EA, IEFAB4F5, IEFAB451.

Exit to: Return to caller.

Called Routines: IDACAT12, IEFAB4EF, IEFDB400 (via SVC99).

IEFAB4F5 — Allocate Catalog Control

Operation: This module can:

- Allocate a catalog.
- Cause an ACB to be created and opened.
- Cause a PCCB to be created and chained or modified.

Entry from: IEFAB469, IEFICATL (an initiator module, IGG0CLCA (a catalog management module), IGG0CLC9 (a catalog management module).

Exit to: Return to caller.

Called Routines: IDACAT11, IEFAB4EA, IEFAB4EF, IEFAB4F4.

IEFAB4F6 — GSPACE/FSPACE Service Routines

Operation: This module obtains 4K blocks of storage and subdivides them as storage requests are made. Requests for release of the 4K blocks and the obtained sections of them are also serviced.

Entry from: Most allocation/unallocation modules, that is, modules with names beginning with IEFAB, IEFBB, or IEFDB.

Exit to: Return to caller.

Called Routines: None.

IEFAB4F7 — SWA Manager Interface

Operation: This module invokes the SWA Manager to assign, read, write, or delete blocks of the Scheduler Work Area and returns the necessary parameters to the caller.

Entry from: IEFAB4A4, IEFAB4A6, IEFAB4DC, IEFAB4EB, IEFAB4FE, IEFAB452, IEFAB453, IEFAB455, IEFAB457, IEFAB458, IEFAB459, IEFAB461, IEFAB464, IEFAB466, IEFBB401, IEFBB402, IEFBB410, IEFBB416, IEFDB4A1, IEFDB400, IEFDB410, IEFDB413, IEFDB414, IEFDB418, IEFDB450.

Exit to: Return to caller.

Called Routines: None.

IEFAB4F8 — Direct Access Label Read

Operation: This module reads direct access volume labels.

Entry from: IEFAB473, IEFAB49B.

Exit to: Return to caller.

Called Routines: None.

IEFAB4F9 — Tape Label Read

Operation: This module reads tape volume labels.

Entry from: IEFAB473.

Exit to: Return to caller.

Called Routines: None.

IEFAB4M4 — Volume Mount & Verify Messages

Operation: This module contains the text used to construct the messages issued by volume Mount and Verify routines.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFAB4M5 — WTO Message Module For AVR and Common Allocation

Operation: This module contains the list forms of WTO and WTOR messages issued by Common Allocation, Automatic Volume Recognition (AVR), and data set reservation/release.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFAB4M6 — Disposition Messages

Operation: This module contains the text used to construct the disposition messages issued by common unallocation.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFAB4M7 — Allocation Messages

Operation: This module contains the allocation messages that are issued to the system output data set and, for unit record devices, to the console.

Entry from: not applicable (non-executable module)

Exit from: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFAB4M9 — Recovery Allocation Messages

Operation: This module contains the text used to construct recovery allocation operator messages.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFAB4UV — Unit Verification

Operation: This routine is invoked by system routines that require information or validity checks based on the contents of the eligible devices table (EDT).

Entry from: IAPINMD, ICBME.

Exit to: Return to caller.

Called Routines: None.

IEFAB421 — Common Allocation Control

Operation: This module is called to allocate one or more data set requests. It is responsible for controlling the following functions:

- Process pending offlines/unloads.
- Initialize work area (ALCWA).
- Allocate DUMMY requests.
- Allocate VIO requests.
- Allocate subsystem requests.
- Determine device requirements.
- Allocate to fixed DA devices.
- Allocate TP requests.
- Volume reservation.
- Allocate by device type.
- Recovery allocation.
- Dequeue from allocation resource.
- Cleanup processing.

Entry from: IEFAB490, IEFBB404, IEFDB413.

Exit to: Return to caller.

Called Routines: IEEAB401, IEFAB4F3, IEFAB422, IEFAB423, IEFAB425, IEFAB427, IEFAB428, IEFAB430, IEFAB431, IEFAB471, IEFAB485, IEFAB49C, IEFAB490, IEFAB491.

IEFAB422 — JES3 Interface

Operation: This module interfaces with the JES3 subsystem, and then updates allocation tables based on whether or not JES3 has made device selections.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: None.

IEFAB423 — Determine Device Requirements

Operation: The function of this module is to determine the device requirements and the device elgibility for each individual request and for the step. Device requirements are represented by the volunit table and device eligibility is represented by the eligible device table (EDT) and eligible device list (EDL). This module builds the volunit table and calls IEFAB424 to build the EDL.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB424, IEFAB426.

IEFAB424 — Build Eligible Device List (EDL)

Operation: This module is responsible for construction of an elgible device list (EDL) for each request.

Entry from: IEFAB423.

Exit to: Return to caller.

Called Routines: IEFAB438.

IEFAB425 — Process TP Requests

Operation: This module controls the processing necessary to allocate teleprocessing requests.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB434.

IEFAB426 — Volunit Affinity Processing

Operation: This module updates volunit table entries to reflect volume affinity. It ensures that every member of a volume affinity has the same unit identification.

Entry from: IEFAB423.

Exit to: Return to caller.

Called Routines: None.

IEFAB427 — Allocate Subsystem Data Sets

Operation: This module controls allocation of subsystem data sets. The major functions performed are:

- Find subsystem requests.
- Issue the IEFSSREQ macro instruction.
- Create DSAB/TIOT entries.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB428.

IEFAB428 — Build/Update/Rebuild Expandable TIOT

Operation: This module creates, updates, or rebuilds the expandable TIOT (DSAB/TIOT DD entry).

Entry from: IEFAB4A3, IEFAB421, IEFAB427, IEFAB431, IEFAB434, IEFAB442.

Exit to: Return to caller.

Called Routines: IEFAB4E9, IEFAB4FC.

IEFAB430 — Fixed Device (Main Path) Control

Operation: This module controls the processing necessary to allocate requests to direct access volumes that are permanently resident or reserved.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB433, IEFAB436.

IEFAB431 — Allocate VIO Data Sets

Operation: This module controls the allocation of VIO data set requests.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB428.

IEFAB432 — Affinity Processor

Operation: This module processes any affinity requests to the volume or unit just allocated. Processing includes ensuring that the affinity requests are valid and, possibly, allocating the requests.

Entry from: IEFAB434.

Exit to: Return to caller.

Called Routines: IEFAB4F2, IEFAB434, IEFAB441.

IEFAB433 — Specific Volume Allocation Control

Operation: This module controls the processing necessary to allocate specific volume requests when the volume is mounted. A specific request is one that requires a particular volume — for example, a volume serial number was specified, the data set was passed, the data set is cataloged.

Entry from: IEFAB430, IEFAB475, IEFAB479, IEFAB485.

Exit to: Return to caller.

Called Routines: IEFAB434, IEFAB442.

IEFAB434 — Allocate Request to Unit

Operation: This module controls the processing necessary to allocate a request to the selected device. Functions include:

- Updating the UCB of the selected device (done by IEFAB435).
- Creating or updating a DSAB and TIOT entry (done by IEFAB428).
- Acquiring space on the volume (interfacing with DADSM).
- Processing requests that specify affinity to the request being handled (done by IEFAB432).

Entry from: IEFAB425, IEFAB432, IEFAB433, IEFAB436, IEFAB441, IEFAB478, IEFAB479, IEFAB489.

Exit to: Return to caller.

Called Routines: IEFAB4F2, IEFAB428, IEFAB432, IEFAB435.

IEFAB435 — Update UCB Routine

Operation: This module updates the UCB to reflect allocation of the current request to it. It also interfaces with VM&V control to unload volumes mounted on the needed unit.

Entry from: IEFAB434.

Exit to: Return to caller.

Called Routines: IEFAB441, IEFAB49C.

IEFAB436 — Nonspecific Volume Allocation Control

Operation: This module controls the processing necessary to allocate non-specific, non-private requests to public and storage volumes. It is responsible for building a list of devices that can satisfy the request, interfacing with the System Resources Manager to select the device to be allocated, and interfacing with IEFAB434 to allocate the request.

Entry from: IEFAB430, IEFAB475, IEFAB485.

Exit to: Return to caller.

Called Routines: IEFAB4F0, IEFAB434, IEFAB440, IEFAB442.

IEFAB438 — Update DDR Count Routine

Operation: This module retrieves the DDR count from the CSD. The count is used by module IEFAB424 to ensure that allocation is synchronized with DDR while the eligible device list (EDL) is being built.

Entry from: IEFAB424.

Exit to: Return to caller.

Called Routines: None.

IEFAB440 — Build Allocated UCB List

Operation: This module builds a list of allocated UCBs (ALCUCBLT) which is used in the interface with the System Resources Manager.

Entry from: IEFAB436, IEFAB478, IEFAB489, IEFAB490.

Exit to: Return to caller..

Called Routines: None.

IEFAB441 — Volume Validity Checker

Operation: This module determines if the input volume is currently mounted. If it is, this module checks to see if the volume is useable by the current request.

Entry from: IEFAB432, IEFAB435, IEFAB486.

Exit to: Return to caller.

Called Routines: IEFAB434, IEFAB442, IEFAB49C.

IEFAB442 — Affinity Remover

Operation: This module cancels unit affinity between the input request and other requests, when the volume being allocated to the input request is permanently resident or reserved.

Entry from: IEFAB433, IEFAB436, IEFAB441, IEFAB479.

Exit to: Return to caller.

Called Routines: IEFAB4F2, IEFAB428.

IEFAB445 — Device Allocation Defaults Module

Operation: This module contains space and unit name defaults used by Device Allocation.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFAB451 — JFCB Housekeeping Control

Operation: This module is responsible for retrieving the information necessary for allocation.

Entry from: IEFBB404, IEFDB413.

Exit to: Return to caller.

Called Routines: IEFAB4EA, IEFAB4F4, IEFAB452.

IEFAB452 — DD Processing Control

Operation: This module selects and initiates the processing of:

- STEPCAT requests.
- Other data set requests.

Entry from: IEFAB451.

Exit to: Return to caller.

Called Routines: IEFAB4EF, IEFAB4F7, IEFAB453, IEFAB454.

IEFAB453 — DD Preparation

Operation: This module does the following processing:

- Initializes the pointer to the JFCB for the requests.
- Processes data set requests that specified QNAME, TERM=TS, DUMMY, DSN=NULLFILE, or PGM=*, stepname.ddname or PGM=*, procstepname.ddname.
- Processes subsystem and VIO data sets.
- Retrieves unit information.

Entry from: IEFAB452.

Exit to: Return to caller.

Called Routines: IEFAB4F7, IEFAB470.

IEFAB454 — DD Function Control

Operation: This module determines what additional information is required for a DD. It then constructs a parameter list and routes control to the appropriate routine(s) to obtain that information.

Entry from: IEFAB452.

Exit to: Return to caller.

Called Routines: IEFAB456, IEFAB457, IEFAB458, IEFAB459, IEFAB469.

IEFAB455 — Passed Data Set Information Scan

Operation: This module scans the PDI (Passed Data Set Information) for the data set indicated. If the data set is found, it returns a pointer (SVAB) to the SIOT that passed the data set.

Entry from: IEFAB469.

Exit to: Return to caller.

Called Routines: IEFAB4EB, IEFAB4F7.

IEFAB456 — Data Set Name Resolution

Operation: This module resolves data set name information and provides volume and unit information for GDG requests.

Entry from: IEFAB454.

Exit to: Return to caller.

Called Routines: IEFAB461, IEFAB466, IEFAB469.

IEFAB457 — Volume/Unit Resolution

Operation: This module receives control for every request in order to resolve volume and unit information for subsequent allocation processing.

Entry from: IEFAB454.

Exit to: Return to caller.

Called Routines: IEFAB4F7, IEFAB463, IEFAB464, IEFAB466, IEFAB469.

IEFAB458 — DCB Resolution

Operation: This module established DCB control information in a JFCB.

Entry from: IEFAB454.

Exit to: Return to caller.

Called Routines: IEFAB4F7, IEFAB469.

IEFAB459 — Disposition Processing

Operation: This module does the following:

- completes data set disposition information in the SIOT
- updates the PDI (passed data set information), if necessary
- marks tape data set requests as private, if necessary
- retrieves catalog information, if necessary

Entry from: IEFAB454.

Exit to: Return to caller.

Called Routines: IEFAB4DC, IEFAB4EB, IEFAB469.

IEFAB461 — GDG Single Processing

Operation: This module converts a GDG (Generation Data Group) data set request from a relative data set name (generation number) to a fully qualified data set name. It obtains volume and unit information for a GDG single request of an existing data set.

Entry from: IEFAB456.

Exit to: Return to caller.

Called Routines: IEFAB4F7, IEFAB469.

IEFAB463 — Multiple Device Type Determination

Operation: This module scans the catalog return information(CRI) to:

- Determine if the data set is a multi-device type data set and, if so, to return the number of different device types minus one.
- Given the number of device types, to find and return the respective device type and a pointer to the volume list entry for that device type.

Entry from: IEFAB457, IEFAB464.

Exit to: Return to caller.

Called Routines: None.

IEFAB464 — Volume Unit Table Completion

Operation: This module copies appropriate volume and unit information from the source determined by previous routines.

Entry from: IEFAB457.

Exit to: Return to caller.

Called Routines: IEFAB4F7, IEFAB463, IEFAB470.

IEFAB466 — Table Creation

Operation: The module creates SIOTs/JFCBs and initializes them.

Entry from: IEFAB456, IEFAB457, IEFAB469.

Exit to: Return to caller.

Called Routines: IEFAB4FD, IEFAB4F7.

IEFAB469 — JLOCATE

Operation: JLOCATE is a Housekeeping Service routine that provides data set name information for GDG single requests and necessary volume and unit information for a given data set name.

Entry from: IEFAB454, IEFAB456, IEFAB457, IEFAB458, IEFAB459, IEFAB461.

Exit to: Return to caller.

Called Routines: IEFAB4EF, IEFAB4F5, IEFAB466.

IEFAB470 — Unit Name Conversion

Operation: This module extracts necessary EDT or UCB device information in the current SIOT.

Entry from: IEFAB453, IEFAB464.

Exit to: Return to caller.

Called Routines: None.

IEFAB471 — Generic Allocation Control

Operation: This module controls the allocation of requests that could not be handled by fixed device allocation (see IEFAB430). It handles requests to any mounted volumes and/or to any online and unallocated units. It processes one generic device type at a time. The order of processing is determined by the installation device precedence list.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB4FA, IEFAB472, IEFAB473, IEFAB475.

IEFAB472 — Build Tables for Generic Processing

Operation: This module builds the tables needed for generic allocation processing. Tables built are:

1) Allocation queue manager request block.
2) Cover/reduce algorithm interface tables.
3) Request identification mask table.
4) Work area for manipulating group masks.

Entry from: IEFAB471.

Exit to: Return to caller.

Called Routines: IEFAB4FA, IEFAB481.

IEFAB473 — AVR (Automatic Volume Recognition) Control

Operation: This module determines if any volumes are mounted on the units whose UCBs are passed as input to this routine and reads the volume label.

Entry from: IEFAB471.

Exit to: Return to caller.

Called Routines: IEFAB4E0, IEFAB4F8, IEFAB4F9, IEFAB493, IEFXVNSL.

IEFAB474 — Process Multi-Unit/Multi-Generic Requests

Operation: This module determines if a multi-unit (multi-generic) request processed by the algorithm is assigned to a single generic device type. If not, this module attempts to find a single generic device type that can satisfy the request.

Entry from: IEFAB476, IEFAB486.

Exit to: Return to caller.

Called Routines: IEFAB481.

IEFAB475 — Allocate Within A Generic

Operation: This module attempts to allocate as many requests as possible to the chosen generic device type. It determines what types of allocation is required and then calls the necessary allocation routines.

Entry from: IEFAB471.

Exit to: Return to caller.

Called Routines: IEFAB433, IEFAB436, IEFAB476, IEFAB479.

IEFAB476 — Allocate Via The Algorithm

Operation: This module processes requests when a choice of units exists: a specific unit was not requested; the request cannot be satisfied by allocating it to a mounted volume; the request cannot be allocated to a permanently resident or reserved volume. The cover/reduce algorithm (IEFAB480) is called to select the device groups from which to allocate.

Entry from: IEFAB475, IEFAB485.

Exit to: Return to caller.

Called Routines: IEFAB474, IEFAB477, IEFAB478, IEFAB480.

IEFAB477 — Unallocate Requests To Be Rearranged

Operation: This module unallocates requests that the algorithm indicates must be rearranged.

Entry from: IEFAB476, IEFAB485, IEFAB486.

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB49C.

IEFAB478 — Allocate From Groups Picked By Algorithm

Operation: This module allocates requests to available (that is, online, unallocated) devices within the device groups selected by the algorithm for the requests.

Entry from: IEFAB476, IEFAB486, IEFAB491.

Exit to: Return to caller.

Called Routines: IEFAB434, IEFAB440, ICBME.

IEFAB479 — Demand Allocation

Operation: This module allocates requests that specified a specific unit address (for example, unit=190) within the generic device type being processed by generic allocation.

Entry from: IEFAB475, IEFAB485, IEFAB491.

Exit to: Return to caller.

Called Routines: IEFAB4F0, IEFAB433, IEFAB434, IEFAB442.

IEFAB48A — Process Offlines/Allocateds

Operation: This module controls the processing necessary:

- To determine whether a request can be satisfied by offline and/or allocated units.
- To inform the operator of the possible options.

Entry from: IEFAB487.

Exit to: Return to caller.

Called Routines: IEFAB4F3, IEFAB488, IEFAB489.

IEFAB480 — Cover/Reduce Algorithm

Operation: This module uses the hungarian algorithm to determine which group or groups of devices should be used to satisfy a generic allocation request.

Entry from: IEFAB476, IEFAB486, IEFAB488, IEFAB489, IEFAB491.

Exit to: Return to caller.

Called Routines: None.

IEFAB481 — Eliminate Ineligible Groups and Generics

Operation: This module marks all generic device types, except the one selected for the current request, as ineligible in the EDL. It also marks group list entries in the algorithm tables as ineligible, when the groups are not in the selected generic device type.

Entry from: IEFAB4F2, IEFAB472, IEFAB474, IEFAB485, IEFAB486.

Exit to: Return to caller.

Called Routines: None.

IEFAB485 — Recovery Allocation

Operation: This module handles allocation requests that could not be allocated by either Fixed Device Control or Generic Allocation. Offline and/or allocated units can be considered during these allocations, if necessary and if allowed by the caller.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB4FA, IEFAB433, IEFAB436, IEFAB476, IEFAB477, IEFAB479, IEFAB481, IEFAB486, IEFAB49C.

IEFAB486 — Offline/Allocated Device Allocation

Operation: This module handles allocation processing when it is necessary to consider offline and/or allocated devices.

Entry from: IEFAB485.

Exit to: Return to caller.

Called Routines: IEFAB4FA, IEFAB441, IEFAB474, IEFAB477, IEFAB478, IEFAB480, IEFAB481, IEFAB487, IEFAB493.

IEFAB487 — Allocation Recovery Interface With Operator

Operation: This module controls the processing necessary to inform the system operator that this allocation cannot proceed without waiting for devices to be unallocated by other jobs and/or bringing needed devices online.

Entry from: IEFAB486.

Exit to: Return to caller.

Called Routines: IEFAB4F3, IEFAB48A, IEFAB488.

IEFAB488 — Recovery Reply Options Processor

Operation: This module builds and issues the appropriate reply options message and processes the operator's reply.

Entry from: IEFAB48A, IEFAB487.

Exit to: Return to caller.

Called Routines: IEFAB4F3, IEFAB480, IEFAB489.

IEFAB489 — Recovery Allocation of Online Devices

Operation: This module determines, for each group of UCBs in the system, whether the number of online units in the group has increased as a result of the system operator's responses to the reply options message. If it has, the routine will attempt to allocate requests to the units in the group.

Entry from: IEFAB48A, IEFAB488.

Exit to: Return to caller.

Called Routines: IEFAB434, IEFAB440, IEFAB480, ICBME.

IEFAB49A — VM&V DOMR & Cleanup Routine

Operation: This module deletes mount messages, if necessary, and releases the mount control blocks used by this allocation when all mount messages have been deleted.

Entry from: IEFAB4E2, IEFAB4E8, IEFAB492, IEFAB496.

Exit to: Return to caller.

Called Routines: IEFAB498.

IEFAB49B — Device End POST Handler

Operation: This module verifies, if necessary, that:

- A device is now ready (that is, a volume has been mounted); or,
- The volume mounted on the device is acceptable in response to a mount message issued by Mount Control or a mount request to the 3850 Mass Storage System.

Entry from: IEFAB496.

Exit to: Return to caller.

Called Routines: IEFAB4F8, IEFAB493.

IEFAB49C — Unload Interface Routine

Operation: This module interfaces with VM&V Control (IEFAB493) to have volumes unloaded and/or rewound. Before calling IEFAB493, this routine invokes IEFAB4E0 (Test Device Ready Routine).

Entry from: IEFAB4A4, IEFAB421, IEFAB435, IEFAB441, IEFAB477, IEFAB485, IEFAB486, IEFAB491, IEFBB416.

Exit to: Return to caller.

Called Routines: IEFAB4E0, IEFAB493.

IEFAB490 — Common Allocation Cleanup

Operation: This module performs cleanup functions when allocation has completed or when a terminating allocation error has been detected.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB4EE, IEFAB4FA, IEFAB4FD, IEFAB4F3, IEFAB421, IEFAB440, IEFAB492.

IEFAB491 — Wait Holding Resources

Operation: This module waits for other job steps to unallocate devices in the device groups from which this allocation needs additional units. When posted because of an unallocation, this routine tries to satisfy the remaining requests.

Entry from: IEFAB421.

Exit to: Return to caller.

Called Routines: IEFAB4FA, IEFAB478, IEFAB479, IEFAB480, IEFAB49C.

IEFAB492 — Allocation/VM&V Interface

Operation: This module interfaces with VM&V Control (IEFAB493) to process volume mount & verify requests (represented by VM&V request blocks build during generic and recovery allocation).

Entry from: IEFAB490.

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB4FA, IEFAB4F0, IEFAB49A, IEFAB493, IEFAB498.

IEFAB493 — Volume Mount and Verify (VM&V) Control

Operation: This module controls unload, mount, and verify processing; it scans VM&V request blocks and calls Unload Control (IEFAB494), Mount Control (IEFAB495), and Verify Control (IEFAB496) to process the requests.

Entry from: IEFAB473, IEFAB49B, IEFAB49C, IEFAB492.

Exit to: Return to caller.

Called Routines: IEFAB494, IEFAB495, IEFAB496.

IEFAB494 — Volume Unload Control

Operation: This module processes all requests to:

- Rewind or rewind and unload tape volumes.
- Unload direct access volumes.

Entry from: IEFAB493.

Exit to: Return to caller.

Called Routines: IEFAB499,IEFAB4E0.

IEFAB495 — Mount Control Routine

Operation: This module builds mount control blocks. For Mass Storage System (MSS) volume requests, this module interfaces with the 3850 Mass Storage System to mount the volumes. For non-MSS volume requests, this module issues mount WTO or WTOR messages.

Entry from: IEFAB493.

Exit to: Return to caller.

Called Routines: IEFAB4E0, IEFAB498, IEFAB499.

IEFAB496 — Verify Control Routine

Operation: This module handles all verify device end and verify label requests by waiting for the specified units to become ready. In the case of verify label, this module also ensures that the requested volume has been mounted.

Entry from: IEFAB493.

Exit to: Return to caller.

Called Routines: IEFAB49A, IEFAB49B, IEFAB498, IEFAB499.

IEFAB498 — MVCA Chain Processor

Operation: This module is invoked to perform one of the following functions:

- Search the MVCA chain for the MVCA for this allocation.
- Add an MVCA to the MVCA chain.
- Delete an MVCA from the MVCA chain.

Entry from: IEFAB4E2, IEFAB4E5, IEFAB4E8, IEFAB49A, IEFAB492, IEFAB495, IEFAB496.

Exit to: Return to caller.

Called Routines: None.

IEFAB499 — VM&V WTO/WTOR Format Routine

Operation: This module builds WTO/R or multiline WTO/R parameter lists for MOUNT and UNLOAD and VERIFY messages using information supplied by the caller.

Entry from: IEFAB494, IEFAB495, IEFAB496.

Exit to: Return to caller.

Called Routines: None.

IEFAB820 — TCTIOT Construction Interface

Operation: This module determines whether SMF accounting functions are required, initializes the TCT storage map, and calls IEFDB4F7 to build the TCTIOT.

Entry from: IEFSD263 via CALL (to IEFSMFAT, alias for IEFAB820).

Exit to: IEFSD263 via branch.

IEFQMWR — Allocation/Termination Communication Area

Operation: This non-executable CSECT contains the address of the nucleus - resident update DDR Count routine (IEFAB438) and a pointer to the first MVCA on the MVCA chain. This area is called the allocation/termination communication area (ATCA) and is mapped by macro IEFZB432.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFBB4M1 — Job Status Messages

Operation: This module contains the message text for the allocation job status operator messages.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFBB4M2 — Allocation Step-Related Messages

Operation: This module contains the message text of allocation step-related error messages that are issued to the system output data set.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called routines: not applicable (non-executable module)

IEFBB4M3 — Allocation DD-Related Messages

Operation: This module contains the message text of the allocation DD-related error messages that are issued to the system output data set.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFBB4M4 — WTO Messages for IEFBB410

Operation: This module contains the message text of the Unallocation job status messages.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFBB4M5 — Unallocation Messages

Operation: This module contains the message text of the unallocation error messages.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFBB401 — Initiator/Allocation Interface

Operation: This module controls the interface between the initiator and step allocation control. (It determines the path that a step or job will take through the initiator.)

Entry from: IEFSD162.

Exit to: Return to caller.

Called Routines: IEEAB401, IEFAB4A6, IEFAB4FC, IEFAB4FD, IEFAB4FE, IEFAB4F3, IEFAB4F7, IEFBB402, IEFBB404.

IEFBB402 — Test EXEC Statement Condition Codes

Operation: This module determines, from the specifications of the COND parameter on the EXEC statement, whether the step should be bypassed or executed.

Entry from: IEFBB401.

Exit to: Return to caller.

Called Routines: IEFAB4FD, IEFAB4F7.

IEFBB404 — Step Allocation Control

Operation: This module receives control when a job step is to be allocated; it controls the following functions:

- JFCB housekeeping
- Common allocation processing
- Reordering and compressing the TIOT
- Writing DD-related error messages

Entry from: IEFBB401.

Exit to: Return to caller.

Called Routines: IEFAB4FC, IEFAB4FD, IEFAB4F3, IEFAB421, IEFAB451.

IEFBB410 — Initiator/Unallocation Interface

Operation: This module provides an interface between the initiator and step unallocation. SIOTs, JFCBs, and JFCBXs are located and chained, end-of-step messages are issued, and step unallocation is invoked.

Entry from: IEFSD164.

Exit to: Return to caller.

Called Routines: IEEAB401, IEFAB4FC, IEFAB4FD, IEFAB4FE, IEFAB4F7, IEFAB412, IEFBB414, IEFBB416, IEFRPREP, IEFTB721.

IEFBB412 — Process Job Condition Codes

Operation: This module compares the return code from the step just executed with the dependency codes specified in the COND parameter on the JOB statement.

Entry from: IEFBB410.

Exit to: Return to caller.

Called Routines: None.

IEFBB414 — Step Unallocation Control

Operation: This module creates the interface to Common Unallocation for the job step being terminated. This involves building a request block for each SIOT on the input SIOT chain and determining the processing required for each of these requests. Allocation messages are also built and issued if requested.

Entry from: IEFAB4DD, IEFBB410.

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB4EE.

IEFBB416 — Job Unallocation

Operation: This module performs disposition processing for passed unreceived data sets and unloads volumes no longer needed by this job.

Entry from: IEFAB4DD, IEFBB410.

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB4EB, IEFAB4F3, IEFAB4F7, IEFAB49C.

IEFDB4A0 — Dynamic Unallocation Control Routine

Operation: IEFDB4A0 handles Dynamic Allocation (SVC99) requests for unallocation. It checks the validity of the request, determines what DSABs should be processed and what processing should be done for them (Unallocation or remove in-use), and invokes the necessary routine(s) to perform the processing.

Entry from: IEFDB400, IEFDB410.

Exit to: Return to caller.

Called Routines: IEFDB4A1, IEFDB4FA, IEFDB4FC, IEFDB4FF, IEFDB460, IEFDB481.

IEFDB4A1 — Dynamic Unallocation Processor

Operation: This module receives control from either Unallocation Control (IEFDB4A0), or Remove In-use Processor (IEFDB481) to perform Unallocation Processing for DSABs pointed to by the input list of DSABPTRs.

Entry from: IEFDB4A0, IEFDB481.

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB4F7, IEFDB4F9.

IEFDB4D0 — DAIR - Dynamic Allocation Interface Routine

Operation: In a TSO environment, this module serves as an interface from various command processors to dynamic allocation.

Entry from: Various command processors and user programs.

Exit to: Return to caller.

Called Routines: IEFDB400 (via SVC 99), IKJEHCIR (via LINK).

IEFDB4FA — Dsname Search Routine

Operation: This module searches the DSABs and JFCBs for the data set name requested by the user. If the dsname is allocated to the current task, the address of its DSAB is returned to the caller.

Entry from: IEFDB4A0, IEFDB470.

Exit to: Return to caller.

Called Routines: None.

IEFDB4FB — Change DDNAME/JES3 Exit

Operation: This module interfaces with the JES3 subsystem whenever a DDNAME or relative position number change has been made by Dynamic Allocation.

Entry from: IEFDB410, IEFDB411, IEFDB450.

Exit to: Return to caller.

Called Routines: None.

IEFDB4FC — Ddname Search Routine

Operation: This module searches the TIOT for a specific ddname. It begins its search at the starting DSAB address passed as input by the caller.

Entry from: IEFDB4A0, IEFDB4FD, IEFDB414, IEFDB450, IEFDB460, IEFDB470, IEFDB490.

Exit to: Return to caller.

Called Routines: None.

IEFDB4FD — Ddname Generation Routine

Operation: This routine generates an eight character ddname.

Entry from: IEFDB410, IEFDB414.

Exit to: Return to caller.

Called Routines: IEFDB4FC.

IEFDB4FE — Obtain DSORG Routine

Operation: This module determines the data set organization for the data set associated with the input DSAB.

Entry from: IEFDB410, IEFDB413, IEFDB470.

Exit to: Return to caller.

Called Routines: None.

IEFDB4FF — Syntax Checker

Operation: This module determines whether the user has entered an invalid key, mutually exclusive keys, or duplicate keys. It also checks whether an invalid number of parameters has been entered with each key, and whether the length of each parameter value is valid.

Entry from: IEFDB4A0, IEFDB412, IEFDB450, IEFDB460, IEFDB470, IEFDD480, IEFDB490.

Exit to: Return to caller.

Called Routines: None.

IEFDB4F8 — Allocate TCTIOT Control

Operation: Following a dynamic allocation, this module updates an existing TCTIOT or obtains and updates a new TCTIOT if the existing TCTIOT cannot be expanded.

Entry from: IEFDB4F9.

Exit to: Return to caller.

Called Routines: None.

IEFDB4F9 — SMF Dynamic DD Routine

Operation: This module updates the TCTIOT to reflect any changes made to the TIOT as a result of a dynamic allocation, unallocation, concatenation, or deconcatenation. If the function is unallocation, concatenation, or deconcatenation, an SMF type 40 record is built and written.

Entry from: IEFDB4A1, IEFDB413, IEFDB450, IEFDB460.

Exit to: Return to caller.

Called Routines: IEFDB4F8.

IEFDB400 — Dynamic Allocation Control Routine

Operation: To process the input request, this module routes control to one of the following functions: dsname allocation, ddname allocation, concatenation, deconcatenation, remove in-use attribute, information retrieval or unallocation.

Entry from: SVC interrupt handler.

Exit to: Return to caller.

Called Routines: IEFAB4FE, IEFAB4F7, IEFDB4A0, IEFDB401, IEFDB410, IEFDB450, IEFDB460, IEFDB470, IEFDB480, IEFDB490.

IEFDB401 — Dynamic Allocation Installation Exit

Operation: The Dynamic Allocation facility of the control program exits to this 'Validation Routine' before doing any processing on behalf of a Dynamic Allocation request. It is entered for all requests, foreground and background. This routine may test and modify the Dynamic Allocation input, and indicate through a return code whether processing is to continue or if the request is to be terminated. This IBM version arbitrarily accepts all requests.

Entry from: IEFDB400.

Exit to: Return to caller.

Called Routines: None.

IEFDB402 — Dynamic Allocation ESTAE Exit

Operation: This module performs clean-up processing when the task under which SVC 99 is executing abnormally terminates.

Entry from: IEFAB4ED.

Exit to: Return to caller.

Called Routines: IEFDB418.

IEFDB403 — TCTIOT FRR Routine

Operation: This exit routine receives control if an ABEND occurred while TCTIOT updating was taking place. This module sets to zero the pointer to the TCTIOT if it is unuseable and frees TCTIOT storage.

Entry from: R/TM (Recovery/Termination Management).

Exit to: Return to caller.

Called Routines: None.

IEFDB410 — Allocate Function Control

Operation: This module is the control routine for the dynamic allocation of a data set.

Entry from: IEFDB400.

Exit to: Return to caller.

Called Routines: IEFAB4F7, IEFDB4A0, IEFDB4FD, IEFDB4FE, IEFDB411, IEFDB412, IEFDB413, IEFDB460.

IEFDB411 — Dynamic Allocation Convert Routine

Operation: This module modifies an existing allocation to satisfy the user's request.

Entry from: IEFDB410.

Exit to: Return to caller.

Called Routines: IEFAB4DC, IEFAB4E9, IEFDB417.

IEFDB412 — Dynamic Allocation Function Validity Checker

Operation: This module validity checks the allocation function's input text against invalid, duplicate, or mutually exclusive keys; invalid number of keyword parameters or parameter lengths; absence of mutually inclusive parameters; invalid parameter values; and improper authorization to request a subsystem data set.

Entry from: IEFDB410.

Exit to: Return to caller.

Called Routines: IEFDB4FF.

IEFDB413 — Normal Dynamic Allocation Control

Operation: This routine controls the processing for a normal dynamic allocation request.

Entry from: IEFDB410.

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB4DC, IEFAB4FC, IEFAB4F7, IEFAB421, IEFAB451, IEFDB4FE, IEFDB4F9, IEFDB414, IEFDB418.

IEFDB414 — Build SWA Tables For Dynamic Allocation Request

Operation: This module creates and sets up a SIOT, JFCB, and if necessary, JFCBX(s).

Entry from: IEFDB413.

Exit to: Return to caller.

Called Routines: IEFAB4F7, IEFDB4FD, IEFDB4FC, IEFDB417.

IEFDB417 — Dynamic Allocation JFCB - DCB Field Update

Operation: This module updates fields in the JFCB for DCB keys specified in the user's request.

Entry from: IEFDB411, IEFDB414.

Exit to: Return to caller.

Called Routines: None.

IEFDB418 — Dynamic Allocation Normal Error Subroutine

Operation: This module serves as a subroutine for clean up processing if an error occurs in dynamic allocation. It releases resources acquired during processing of a dynamic allocation request and ensures that the 'WRITE-EPA' chain is written to SWA.

Entry from: IEFDB402, IEFDB413.

Exit to: Return to caller.

Called Routines: IEFAB4A0, IEFAB4DC, IEFAB4F7.

IEFDB450 — Concatenation Routine

Operation: This module concatenates existing allocations in the order in which they are specified in the input text units.

Entry from: IEFDB400.

Exit to: Return to caller.

Called Routines: IEFAB4FC, IEFAB4F7, IEFDB4FC, IEFDB4FF, IEFDB4F9.

IEFDB460 — Deconcatenation Routine

Operation: This module disassociates members of a dynamically concatenated group by restoring the pre-concatenation ddnames to the members and removing the 'Dynamically Concatenated' (DSABDCAT) and 'Member of a Concatenated Group' (DSABCATM) attributes from the respective DSABs.

Entry from: IEFDB4A0, IEFDB400, IEFDB410.

Exit to: Return to caller.

Called Routines: IEFAB4FC, IEFAB4F7, IEFDB4FC, IEFDB4FF, IEFDB4F9.

IEFDB470 — Dynamic Information Retrieval

Operation: This module provides the user with information about his current allocation environment.

Entry from: IEFDB400.

Exit to: Return to caller.

Called Routines: IEFDB4FA, IEFDB4FC, IEFDB4FE, IEFDB4FF.

IEFDB480 — Remove In-Use Control Routine

Operation: This module handles Dynamic Allocation (SVC 99) requests for Remove In-Use processing. It validity checks the request, determines what DSABs should be processed, and invokes the Remove In-Use Processor (IEFDB481).

Entry from: IEFDB400.

Exit to: Return to caller.

Called Routines: IEFDB4FF, IEFDB481.

IEFDB481 — Remove In-Use Processor

Operation: This module performs Remove In-Use Processing for the DSABs pointed to by the input list.

Entry from: IEFDB4A0, IEFDB480.

Exit to: Return to caller.

Called Routines: IEFDB4A1.

IEFDB490 — Ddname Allocation

Operation: This module assigns the In-Use attribute to the resource associated with the specified ddname.

Entry from: IEFDB400.

Exit to: Return to caller.

Called Routines: IEFDB4FC, IEFDB4FF.

IEFDPOST — Device Interrupt Routine

Operation: This module is invoked by IOS to post tasks waiting for devices to become ready.

Entry from: IOS (Input/Output Supervisor)

Exit to: Return to caller.

Called Routines: IEAOPT01.

IEFDSLST — Data Set Enqueue Parameter List Builder

Operation: This module builds a data set enqueue parameter list for each job; to do this, it uses input passed by IEFDSTBL.

Entry from: IEFSD161 via branch.

Exit to: IEFSD161 via branch.

IEFDSTBL — Data Set Tree Builder

Operation: This module eliminates duplicate data set names from the data set enqueue parameter list that exists for each job.

Entry from: IEFSD161 via branch.

Exit to: IEFSD161 via branch.

IEFIB600 — Initiator/Subsystem SWA Interface

Operation: This module builds the JSCB chain for a job, invokes the interpreter, and builds a CSCB for a job that is not a started task or LOGON.

Entry from: The master subsystem or the job entry subsystem via LINK.

Exit to: Caller via branch.

IEFIB605 — SWA Reconstruction Module

Operation: This module reconstructs SWA for restarted jobs by invoking the SWA merge routine and the data set description record processor.

Entry from: IEFIB605 via CALL.

Exit to: Caller via branch.

IEFIB620 — ESTAE Exit Routine

Operation: Depending upon what type of error caused this module to receive control, it records the error and then either takes a dump or prepares for a retry.

Entry from: IEAVTAS1 via a SYNCH macro instruction.

Exit to: IEAVTAS1 via branch.

IEFIB621 — Initiator Task Recovery Retry Routine

Operation: This module attempts to resume normal initiator processing after a program check, an ABEND, or machine check has occurred, or after an operator has pushed the RESTART key.

Entry from: This module receives control and executes as a result of an RB (request block) created by the recovery termination management routines.

Exit to: IEFSD164 via branch.

IEFIB645 — SWA Create Exit Routine

Operation: This module receives control whenever an error has occurred during interpreter, restart, or SWA create interface processing. It records the error on the system error record data set and then attempts a retry.

Entry from: Recovery/termination management via LINK.

Exit to: Return to caller via branch.

IEFIB650 — Initiator Message Module

IEFIB660 — Build TCTIOT Routine

Operation: This module constructs a TCTIOT based on the current TIOT. If a TCTIOT exists on entry to this module, it is freed. This situation could occur if dynamic allocation obtained a TCTIOT prior to initiation.

Entry from: IEFAB820.

Exit to: Return to caller.

Called Routines: None.

IEFICATL — Initiator Interface to Allocate Catalog Control

Operation: This module determines the data set names of each catalog to be used by a jobstep/task and then invokes IEFAB4F5 to open the catalogs.

Entry from: IEFSD162 via BALR.

Exit to: IEFSD162 via branch.

IEFICPUA — Assignment of CPU Task Affinity

Operation: This module indicates to the calling module which CPUs can be used to run a jobstep/task; it also notifies the calling routine if the required CPU(s) is not on line.

Entry from: IEFSD101 for the first jobstep in a job and from IEFSD101 for subsequent steps.

Exit to: The calling module via branch.

IEFIIC — Initiator Interface Control

Operation: This module puts the initiator routines into supervisor state and builds the initiator's SSIB) SSOB header, entrance, options, and exit list (IEL).

Entry from: IEFSD263 via ATTACH.

Exit to: IEFSD060 via branch.

IEFIMASK — Conversion of Bit Mask

Operation: This routine converts bit positions in a string to hexidecimal characters for inclusion in some initiator messages.

Entry from: IEFSD101 or IEFSD161 via branch.

Exit to: The calling module via branch.

IEFIRECM — Initiator Resource Manager

Operation: This module cleans up resources used by the initiator in an address space which is terminating. This routine deletes all CSCBs associated with this address space from the CSCB chain.

Entry from: RTM (via address in CVTIRECM field).

Exit to: RTM.

IEFISEXR — Initiator Subsystem ESTAE Exit Routine

Operation: This module receives control after an abnormal situation occurs in the initiator or Subsystem Interface Resource Managers. It issues a DEQ for the CSCB chain and makes an error entry in the LOGREC data set.

Entry from: RTM (via address in CVTJRECM field).

Exit to: RTM.

IEFI922B — Builder of 922 Completion Code Interface

Operation: This routine builds an interface to module IEFSD164 to enable the termination of a jobstep/task which has already successfully completed allocation processing.

Entry from: IEFIB621 via CALL.

Exit to: IEFIB621 via branch.

IEFJACTL — Pseudo Access Method Control

Operation: This routine, a part of the master subsystem, provides a data manipulation service for the converter and interpreter at a time when no access method services are available via the RPL/ACB interface.

Entry from: From the converter or interpreter.

Exit to: Return to caller.

Error exit: To caller's ESTAE routine with user ABEND code 0B1 or 0B3.

IEFJCDLT — Storage Deletion Routine

Operation: This routine, a part of the master subsystem, frees the storage no longer needed by the master subsystem after creating SWA control blocks for a task.

Entry from: From IEFJJOBS.

Exit to: Return to caller.

IEFJCNTL — JCLS to SWA Conversion

Operation: This routine, a part of the master subsystem, converts a JCLS chain to SWA control blocks by invoking the converter, then the interpreter (via the SWA-create interface).

Entry from: From IEFJJOBS.

Exit to: Return to caller.

Error exit: To caller's ESTAE routine with user ABEND code 0B1, 0B4, or 0B5.

IEFJDIRD — Pseudo Access Method Direct Read

Operation: This routine, a part of the master subsystem, moves the specified record into a specified buffer so that the record may be updated.

Entry from: From the converter via the RPL/ACB interface.

Exit to: Return to caller.

IEFJDSNA — Data Set Name Assignment

Operation: This routine, a part of the master subsystem, assigns a data set name to each SYSOUT data set specified in the master JCL or in the JCL used to start a job entry subsystem.

Entry from: From allocation.

Exit to: Return to caller.

IEFJDWRT — Pseudo Access Method Direct Write

Operation: This routine, a part of the master subsystem, overlays an old record with an updated record from a buffer.

Entry from: From the converter via the RPL/ACB interface.

Exit to: Return to caller.

IEFJJCLS — JCL to JCLS Conversion

Operation: This routine, a part of the master subsystem, converts the master scheduler JCL (MSTRJCL) to a JCLS chain.

Entry from: From IEFJJOBS.

Exit to: Return to caller.

Error exit: To caller's ESTAE routine with user ABEND code 0B1.

IEFJJOBS — Subsystem Initiation

Operation: This routine, a part of the master subsystem, controls the creation of SWA control blocks when a task is being started via the master subsystem (that is, when the master scheduler or a job entry subsystem is being started).

Entry from: From the initiator via the subsystem interface.

Exit to: Return to the initiator.

Error exit: To caller's ESTAE routine with user ABEND code 0B1.

IEFJJTRM — Subsystem Job Termination

Operation: This routine, a part of the master subsystem, is a dummy routine which, when the job being terminated is a subsystem, replaces normal job termination function.

Entry from: From a system routine via the subsystem interface.

Exit to: Return to the system routine.

IEFJRASP — Common Request Router

Operation: This routine, a part of the master subsystem, handles requests that require processing from all active subsystems except the master subsystem.

Entry from: From a system routine via the subsystem interface.

Exit to: Return to the system routine.

IEFJREAD — Pseudo Access Method Sequential Read

Operation: This routine, a part of the master subsystem, moves a record from a chain of records to a buffer and prepares for the next record to be read from the chain.

Entry from: From the converter or interpreter via the RPL/ACB interface.

Exit to: Return to the caller.

IEFJRECM — Subsystem Interface Resource Manager

Operation: This module checks the resource manager parameter list for task or address space termination indicators. It then builds either a task or address space SSOB extension. If an address space is terminating, it also builds a list of jobnames associated with the terminating address space. It then issues IEFSSREQ macro to communicate end-of-task or end-of-address space to all active subsystems.

Entry from: RTM (via address in CVTJRECM field).

Exit to: RTM.

IEFJSDTN — Subsystem Determination

Operation: This routine, a part of the master subsystem, determines if the task being started is a subsystem.

Entry from: From the initiator via the subsystem interface.

Exit to: Return to the initiator.

IEFJSREQ — Subsystem Interface

Operation: IEFJSREQ handles requests for services from a job entry subsystem or the master subsystem. It determines which routine of which subsystem is to receive control and then passes control to that routine via a branch instruction.

Entry from: From a system routine via the IEFSSREQ macro instruction.

Exit to: To the subsystem routine to perform the requested function. (see "Subsystem Interface," Section 2, for module names.)

Error exit: Return to caller with non-zero return code.

IEFJWRTE — Pseudo Access Method Sequential Write

Operation: This routine, a part of the master subsystem, adds a record to a chain of records by obtaining storage, moving the new record to that storage area, and chaining the previous record to the new one.

Entry from: From the converter via the RPL/ACB interface.

Exit to: Return to the converter.

IEFJWTOM — Subsystem Initiation Message Writer

Operation: This routine, a part of the master subsystem, issues to hardcopy the converter, interpreter, and allocation error messages (including the JCL card images in error) for tasks being started via the master subsystem.

Entry from: From the converter, interpreter, or allocation via the RPL/ACB interface.

Exit to: Return to caller.

IEFNB901 — Data Management Interpreter Exit Routine

Operation: This module builds a parameter list for the data management interpreter routine and then passes control to that module which processes the AMP keyword parameter.

Entry from: From IEFVDA via branch.

Exit to: To data management interpreter routine via LOAD and BALR; to caller via branch.

IEFNB903 — Interpreter Initialization Routine

Operation: This routine obtains storage for and partially initializes the interpreter work area (IWA), its I/O buffer, the JMR, and the DSENQ table. It also establishes an ESTAE environment for the interpreter.

Entry from: From IEFIB600 via LINK.

Exit to: To IEFVHE via branch or to the journal write routine via branch for restarted jobs.

IEFQB550 — SWA Manager Move Mode Functions

Operation: This module assigns, writes, reads, or deletes scheduler work area (SWA) blocks in move mode.

Entry from: Initiator/terminator, Scheduler Restart, converter/interpreter, IEFQB580 or IEFQB585.

Exit to: Caller via branch.

IEFQB555 — SWA Manager Locate Mode Functions

Operation: This module assigns, writes, reads, or deletes scheduler work area (SWA) blocks in locate mode.

Entry from: Initiator/terminator or Scheduler Restart or allocation via SWAREQ macro instruction.

Exit to: Caller via branch.

IEFQB580 — QMNGRIO Macro Interface Handler

Operation: This module provides an interface to SWA manager functions from data management routines that use the QMNGRIO macro instruction. IEFQB580 builds and initializes a queue management parameter area (QMPA) on behalf of the module requesting SWA manager functions.

Entry from: Data management routines via QMNGRIO macro instruction.

Exit to: IEFQB550 via BALR.

IEFQB585 — SWA Manager Interface Module

Operation: This module intercepts any references to previously existing queue manager modules. IEFQB585 inserts the appropriate function code in the queue management parameter area (QMPA) on behalf of the requesting module.

Entry from: Modules that reference previously existing queue manager modules.

Exit to: IEFQB550 via BALR.

IEFRPREP — Restart Preparation Routine

Operation: This module determines if an ABENDed jobstep task can be restarted.

Entry from: IEFBB410

Exit to: IEFBB410

IEFSDPPT — Program Properties Table

IEFSD060 (IEFSD160) — Initiator Control Initialization

Operation: This module builds the linkage control block (LCT) for a jobstep/task to be processed by the initiator and a STAE parameter list (STEPL) for the initiator's STAE exit routine.

Entry from: IEESB605 via LINK for LOGON, START, and MOUNT commands; from IEFIIC via branch for normal initiator processing; from IEEVIPL via ATTACH for master scheduler initialization.

Exit to: IEFSD061 via branch.

IEFSD061 (IEFSD161) — Job Select Routine

Operation: This routine requests jobs for the initiator from the job entry subsystem and begins STOP processing as required for started tasks. For a selected job that is to be warmstarted, IEFSD061 sets up for warmstart processing; for all other selected jobs this module assigns special protect keys and processes data set integrity, if it is required.

Entry from: IEFSD160 via branch for job selection or from IEFSD166 for job selection or stop processing.

Exit to: A generalized initiator exit via branch, LINK, or XCTL for internal stop processing; to IEFSD164 via branch for warmstart; to IEFSD101 for normal initiator processing.

IEFSD062 (IEFSD162) — Device Allocation Interface Routine

Operation: This module invokes device allocation for a jobstep/started task, opens a JOBLIB, STEPLIB, or FETCHLIB required by the jobstep/started task, and completes assignments of special properties.

Entry from: IEFSD102 via branch.

Exit to: IEFSD103 via BALR, to IEFSD164 via BALR if an error occurred during allocation or OPEN catalog processing.

IEFSD064 (IEFSD164) — Step Delete Routine

Operation: When a jobstep/task has completed processing, this module closes the JOBLIB, STEPLIB, or FETCHLIB DCBs, performs job and jobstep timing calculations, and builds a dummy TCB to be used by termination routines.

Entry from: from IEFSD161 via branch for warmstarted jobs; from IEFSD162 via branch when an error has occurred in allocation for a jobstep/started task; from IEFSD263 via branch when a jobstep/started task has completed processing,and from IEFIB621 to retry after ESTAE processing.

Exit to: IEFSD101 via branch to initiate the next step in a job; to IEFSD166 via branch to delete a job when all jobsteps have completed or to suspend a job, if necessary.

IEFSD066 (IEFSD166) — Job Delete Routine

Operation: This module deletes a job and its associated control blocks when the job has completed processing or re-enqueues a job when necessary.

Entry from: IEFSD164 via branch.

Exit to: IEFSD161 via branch.

IEFSD101 — PPT Scan

Operation: This routine scans the program properties table (PPT) and assigns special properties to a jobstep/started task as indicated; it also builds a GETPART work table if one is required by the jobstep/started task.

Entry from: IEFSD161 via branch for the first jobstep in job and from IEFSD164 via branch for subsequent jobsteps.

Exit to: IEFSD102 via branch.

IEFSD102 — Data Set Enqueue

Operation: This routine enqueues on the data sets required for an entire job.

Entry from: IEFSD101 via branch.

Exit to: IEFSD162 via branch.

IEFSD103 — ATTACH Interface Routine

Operation: This module builds the ATTACH parameter list.

Entry from: IEFSD162 via branch.

Exit to: IEFSD263 via branch.

IEFSD160 — See IEFSD060

IEFSD161 — See IEFSD061

IEFSD162 — See IEFSD062

IEFSD164 — See IEFSD064

IEFSD166 — See IEFSD066

IEFSD263 — Initiator ATTACH Module

Operation: This module gets a region for a jobstep/started task, if one is required, attaches the jobstep/started task, and then waits for an end-of-task or cancel ECB to be posted. When the appropriate ECBs are posted, this module detaches the jobstep/started task.

Entry from: IEFSD103 via branch.

Exit to: IEFSD164 via branch.

IEFSMFIE — SMF Initialization Exit Support Module

Operation: This module constructs a timing control table (TCT) and supports user's job initiation exit routines and step initiation exit routines. It also issues SMF job commencement record type 20.

Entry from: IEFSD101 via BALR.

Exit to: IEFSD101 via branch.

IEFTB721 — SMF Unallocation Control Routine

Operation: This module performs SMF processing at step and job unallocation. It involves IEFACTRT.

Entry from: IEFBB410 via CALL.

Exit to: IEFBB410 via branch.

IEFTB722 — SMF Record Construction at Unallocation

Operation: This module creates SMF records for step and job termination for batch and TSO users. It also issues messages to the programmer with information on step and job times.

Entry from: IEFTB721 via CALL.

Exit to: IEFTB721 via branch.

IEFTB723 — SMF Exit Write-to-Programmer Processor

Operation: This module provides an interface that can be used by IEFACTRT accounting exits to issue messages to the programmer.

Entry from: May be called via BALR from IEFACTRT. (IEFYS is an alternate entry name for IEFTB723).

Exit to: IEFACTRT via branch.

IEFUJI — User Job Initiation Exit Routine

IEFUJV — Converter/Interpreter SMF User Exit Routine

IEFUSI — User Step Initiation Exit Routine

IEFUTL — User Time Limit Exit Routine

IEFVDA — Interpreter DD Card Processor

Operation: This routine creates and initializes the control blocks for a DD statement. It also checks the validity of DD keyword values and then sets a job fail indicator if it finds an invalid keyword value.

Entry from: IEFVHE via branch.

Exit to: IEFVHE via branch.

IEFVDBSD — Interpreter DSENQ Table Processor

Operation: This module creates a DSENQ table for all explicitly named (by DSNAME parameter) data sets and marks each as exclusive or shared.

Entry from: IEFVDA or IEFVEA via branch.

Exit to: Caller via branch.

IEFVEA — Interpreter EXEC Statement Processor

Operation: This routine creates a step control table (SCT) for an EXEC statement and the override and refer-back tables required for the step. It also chains the SIOTs and JFCBs for a job library, job catalog, and/or concatenation of data sets for the step.

Entry from: IEFVHE via branch.

Exit to: Caller via branch.

IEFVFA — Converter Scan Routine

Operation: This module scans the JCL card images for syntax errors and merges JCL from the JCL data set and a cataloged or in-stream procedure if required. It then converts each JCL statement into internal JCL text.

Entry from: IEFVHEB via branch.

Exit to: IEVHF via branch.

IEFVFB — Converter Symbolic Parameter Routine

Operation: This module assigns values specified on an EXEC procedure statement or PROC statement to symbolic parameters that appear on JCL statements in cataloged or in-stream procedures.

Entry from: IEFVFA via BALR.

Exit to: Caller via branch.

IEFVGK — Interpreter GET Key/Positional Utility Routine

Operation: This routine sets a pointer to the next JCL keyword or positional parameter in the internal text buffer for the JCL statement processor routines, IEFVJA, IEFVEA, or IEFVDA.

Entry from: IEFVJA via branch, or from IEFVEA via branch, or from IEFVDA via branch.

Exit to: Caller via branch.

IEFVGM — Converter/Interpreter Message Module

Operation: This module puts error messages for the entire converter/interpreter into the message data set, and/or any required JCL statements to the list data set.

Entry From: IEFVJA via branch, or from IEFVEA via branch, or from IEFVDA via branch.

Exit to: Caller via branch.

IEFVGT — Converter Test and Store Utility Routine

Operation: This module performs miscellaneous functions for the JCL statement processors, IEFVJA, IEFVEA, and IEFVDA, based on indicators in the parameter descriptor table passed to it by the processor routines.

Entry from: IEFVJA, IEFVEA, or IEFVDA via branch.

Exit to: Caller via branch.

IEFVHA — Converter GET Routine

Operation: This module reads into an input buffer JCL statements from the JCL data set, from the procedure library, and/or from in-stream procedures.

Entry from: IEFVHC or IEFVIND via branch.

Exit to: IEFVHC via branch.

IEFVHC — Converter Comment or Continuation Validation Routine

Operation: This routine determines whether a valid comment or continuation is indicated on a JCL statement.

Entry from: IEFVHA via branch.

Exit to: to IEFVHEB via branch, if a continuation was expected and was or was not received; to IEFVHA via branch, if a comment was received; to IEFVHCB, via branch, if no continuation was expected.

IEFVHCB — Converter Verb Identifier Routine

Operation: This routine identifies the verb on a JCL statement and merges JCL statements from the JCL data set and the procedure library.

Entry from: IEFVHC, IEFVHL, or IEFVHF via branch.

Exit to: to IEFVHA, via branch, when the JCL statement does not contain a verb; to IEFVHA, via branch, when the procedure library buffer must be primed; to IEFVHEB, via branch, when the JCL statement contains a JOB, EXEC, DD, or NULL verb; to IEFVHM, via branch, when the verb is unrecognizable; to IEFVINA, via branch, when a PROC verb is found.

IEFVHE — Interpreter GET and Route Routine

Operation: This routine gets JCL text strings from the internal text data set and routes them to the appropriate processor. It also determines when the SCT for a step or the JCT for a job should be written into SWA.

Entry from: IEFNB903 via branch.

Exit to: to IEFVJA via branch when a JOB statement text string is encountered; to IEFVEA via branch when an EXEC statement text string is encountered; to IEFVDA via branch when a DD statement text string is encountered; to IEFVHH when an SCT or JCT is to be written into SWA.

IEFVHEB — Converter Pre-scan Routine

Operation: This routine performs checkpoint/restart functions, enters JCL statements into the list data set, and initializes the JMR for SMF processing.

Entry from: IEFVHC or IEFVHCB via branch.

Exit to: When IEFVHEB is used as a subroutine, it returns to its caller via branch; when it has been used to initialize a JMR, or to process a SYSCHK DD statement or a PROC statement, it branches to IEFVEA.

IEFVHF — Converter Termination Routine

Operation: This module frees storage space used by the converter, issues messages to the operator about the status of the job after converter processing, initializes SYSCHK DD statement processing and deactivates the converter ESTAE environment.

Entry from: IEFVFA via branch.

Exit to: IEFVHA when continuation of JCL text is expected; to IEFVHCB when no continuation is expected or to continue SYSCHK DD processing; to IEFVHEB if a procedure statement is being overridden.

IEFVHH — Interpreter ENQUEUE Routine

Operator: This module writes job and step tables into SWA as required and then performs miscellaneous clean-up processing for the interpreter.

Entry from: IEFVHE via branch.

Exit to: IEFVHN via branch.

IEFVHL — Converter NULL Statement or END-OF-FILE Processor

Operation: This routine performs end-of-file processing for JCL statements in the JCL data set or in a cataloged or in-stream procedure.

Entry from: IEFVHEB via branch.

Exit to: IEFVHA, via branch, to continue reading a procedure; to IEFVHEB, via branch to continue processing a JCL statement from a procedure; to IEFVHCB, via branch, to continue reading JCL statements from the JCL data set; to IEFVHF, via branch, when both the JCL data set and all procedures are completed.

IEFVHM — Converter Command Verb Validation Routine

Operation: This routine determines if a valid command has been specified on a JCL statement.

Entry from: IEFVHCB via branch.

Exit to: IEFVHA via branch.

IEFVHN — Interpreter Termination Routine

Operation: This module frees storage space used by the interpreter, issues messages to the operator about job status at the end of interpreter processing, and deactivates the ESTAE environment over the interpreter.

Entry from: IEFVHH via branch.

Exit to: The interpreter's original caller, SWA create interface, IEFIB600, via branch.

IEFVHQ — Converter SWA Manager Interface

Operation: This routine is the interface between either the converter or interpreter and the SWA manager and between the interpreter and the journal merge routine.

Entry from: IEFVDA, IEFVEA, IEFVFB, IEFVHA, IEFVHH, IEFVINA, or IEFVJA)A via branch.

Exit to: Return to caller via branch.

IEFVHR — Converter/Interpreter Operator Message Module

Operation: This module puts converter and interpreter error messages for the operator into the message data set.

Entry from: IEFVHA, IEFVHF, IEFVHM, or IEFVHN via branch.

Exit to: Return to caller via branch when recovery is possible; to IEFVHN, via branch, if recovery is not possible.

IEFVH1 — Converter Initialization Routine

Operation: This routine performs initialization processing for the converter, obtains storage space for work areas and buffers, and also establishes the ESTAE environment over the converter.

Entry from: The job entry subsystem or the master subsystem.

Exit to: IEFVHA via branch.

IEFVINA — Converter In-stream Procedure Control and GTF Routine

Operation: This module reads in-stream procedures from the JCL data set, records each procedure, and saves each one in compressed form in the scheduler work area (SWA).

Entry from: IEFVHCB via branch.

Exit to: IEFVHA, via branch, for each successive statement in an in-stream procedure.

IEFVINB — Converter In-stream Procedure Directory Search Routine

Operation: This routine searches through the procedure directory for a procedure name specified on a JCL statement. When it locates the procedure name, it returns to the calling routine the address at which that procedure begins.

Entry from: IEFVINA via branch.

Exit to: Caller via branch.

IEFVINC — Converter In-stream Procedure Directory Build Routine

Operation: This module builds an entry for a specified procedure in the in-stream procedure directory.

Entry from: IEFVINA via branch.

Exit to: Caller via branch.

IEFVIND — Converter In-stream Procedure Expand Interface Routine

Operation: This module functions as an access method for IEFVHA. Whenever IEFVHA issues a READ macro instruction for a record from an in-stream procedure, IEFVIND sets up required parameter lists and then invokes IEZDCODE to expand the record.

Entry from: IEFVHA via branch.

Exit to: IEZDCODE, via branch, to expand a record, to IEFVHQ, via branch, to read another record; to caller, via branch, when processing has completed.

IEFVINE — Converter In-stream Procedure Syntax Checker

Operation: This routine checks the label field of a PROC or PEND statement appearing in an in-stream procedure.

Entry from: IEFVINA via branch.

Exit to: Caller via branch.

IEFVJA — Interpreter JOB Statement Processor

Operation: This module creates and initializes the control blocks and tables required by a JOB statement and also checks the validity of JOB statement keyword values.

Entry from: IEFVHE via branch.

Exit to: Caller via branch.

IEFVKMSG — Test EXEC Dependency Codes Message

Operation: This module contains the message text of the message issued by IEFBB402, in the event that a step is not run because of the COND parameter on the EXEC statement.

Entry from: not applicable (non-executable module)

Exit to: not applicable (non-executable module)

Called Routines: not applicable (non-executable module)

IEFXB500 — Journal Write Routine

Operation: This module writes 'critical' control blocks to the job journal for purposes of restart or termination.

Entry from: SWA Manager, Initiator, VIO.

Exit to: Caller.

IEFXB601 — Journal Merge Routine

Operation: This module merges the control blocks from the job journal to the scheduler work area (SWA) to establish a pre-job failure-environment.

Entry from: IEFIB605.

Exit to: IEFIB605.

IEFXB602 — Restart Interface Routine

Operation: This module builds a virtual address table (VAT) to be used when the journal merge routine reconstructs SWA.

Entry from: IEFVHQ.

Exit to: IEFVHQ.

IEFXB603 — Automatic/Checkpoint Restart Message Module

Operation: This module contains message texts for journal service and checkpoint routines. The module is non-executable.

Entry from: IEFXB500, IEFRPREP, IEFXB609, IEFXB601.

Exit to: Caller.

IEFXB604 — Step Header Create

Operation: Except for automatic step or automatic checkpoint restarts, this module builds the step header record for each job step, before putting anything else in the job journal for the step.

Entry from: IEFSD162.

Exit to: IEFSD162.

IEFXB609 — Data Set Descriptor Record Processor

Operation: This module makes the SWA entries for restarting jobs reflect the job environment at the time a checkpoint was issued.

Entry from: IEFIB605.

Exit to: IEFIB605.

IEFXB610 — Open Interface Module

Operation: This routine opens the checkpoint data set.

Entry from: IEFXB609.

Exit to: IEFXB609.

IEFXVNSL — User-Replaceable Non-Standard Label Routine

Operation: This user-replaceable module returns a code of 4 if it is called because no user non-standard label handling routine exists to process an NSL label volume.

Entry from: IEFAB473.

Exit to: Return to caller.

Called Routines: None.

IEWSUOVR — Overlay Supervisor Resident Module

Operation: This module performs overlay requests.

Entry from: Overlay segment.

Exit to: Overlay supervisor processor IEWSWOVR.

IEWSWOVR — Overlay Supervisor Processor

Operation: Performs the overlay request and causes segments to be brought into virtual storage.

Entry from: Overlay supervisor resident module IEWSUOVR.

Exit to: Overlay segment.

IEZDCODE — Interpreter In-stream Procedure Decompression Subroutine

Operation: This module re-expands the in-stream procedure records that were compressed by IEZNCODE.

Entry from: IEFVIND via branch.

Exit to: Caller via branch.

IEZNCODE — Interpreter In-stream Procedure Compression Subroutine

Operation: This module compresses the data records that comprise an in-stream procedure by replacing blanks with count fields. It then blocks the compressed records.

Entry from: IEFVINA via branch.

Exit to: Caller via branch.

IGC0Z03F (IEEJB840) — Write-to-Programmer

Operation: Process Write-to-Programmer requests (messages) for WTO and WTOR macro instructions with routing code 11.

Entry from: IEAVVWTO.

Exit to: IEAVVWTO.

IGF2503D — SWAP Command Processor

Operation: This module processes all SWAP commands.

Entry from: IEE0403D.

Exit to: Caller; IEE0503D.

IGF2603D — MODE Command Processor

Operation: This module processes all MODE commands.

Entry from: IEE0403D.

Exit to: Caller; IEE0503D.

IGX00013 — MFSTART Mainline Processor

Operation: Controls the initialization and termination of routines that perform MF/1 functions.

Entry from: IGC0010I.

Exit to: IGC0010I.

IGX00014 — MFDATA SVC Mainline Processor

Operation: Controls the operation of measurement gathering routines that operate once each interval.

Entry from: IGC0010I.

Exit to: IGC0010I.

IKJEES20 — SEND Command Message Module

Operation: This module contains the messages for the SEND command processor.

Entry from: IEEVSEND; IKJEES10; IKJEES40.

Exit to: Caller.

IKJEFLA — LOGON Initialization

Operation: This routine initializes the control blocks necessary for LOGON processing: LWA, PSCB, JSEL, UPT, RLGB, and JSXL.

Entry from: IEEPRWI2 (STC).

Exit to: IKJEFLB.

Error exit: IEEPRTN.

IKJEFLB — LOGON Scheduler

Operation: This routine invokes the initiator to schedule a TSO terminal session and performs certain functions for the LOGON prompting monitor.

Entry from: IKJEFLA.

Exit to: IEESB605.

Error exit: IEEPRTN.

IKJEFLC — LOGON Prompting Monitor

Operation: This routine controls the processing flow for the LOGOFF processor, LOGON verification, and the LOGON information routine.

Entry from: IKJEFLB.

Exit to: Return to the system.

IKJEFLCM — Prompting Monitor Message Text

Operation: This is a non-executable module that contains the message text for messages issued by IKJEFLC.

Entry from: not applicable

Exit to: not applicable

IKJEFLD — LOGON Pre-prompt Exit

Operation: This optional routine is written by the installation to perform authorization and accounting at LOGON time.

Entry from: IKJEFLI.

Exit to: Return to IKJEFLI.

IKJEFLE — LOGON Verification

Operation: This routine ensures that the LOGON command entered by the terminal user is valid and prompts the user to supply missing information or to correct invalid information.

Entry from: IKJEFLC.

Exit to: Return to IKJEFLC.

IKJEFLEA — Parse/Scan Interface

Operation: This routine invokes IKJSCAN and IKJPARS to determine if the LOGON command is syntactically valid. It prompts the terminal user to supply required parameters that are missing and to correct invalid parameter values.

Entry from: IKJEFLE.

Exit to: Return to IKJEFLE.

IKJEFLF — System-Initiated Cancel Schedule Routine

Operation: This module schedules an SRB routine and checks the validity of the input to this routine.

Entry from: IEE3703D, TIOC Routines.

Exit to: Caller.

IKJEFLG — Attention Exit

Operation: Following an attention interrupt, this routine issues second level messages, causes a re-logon, and returns control to the interrupted processing.

Entry from: An interrupt handler following an attention interrupt from a terminal.

Exit to: Return to caller.

IKJEFLGB — LOGON Prompter's Recovery

Operation: Analyze the failure and schedule a dump, if necessary. Set up control blocks to attempt a retry of LOGON verification, if possible.

Entry from: ABEND processing for the LOGON monitor task.

Exit to: IKJEFLC for retry or continue with ABEND.

IKJEFLGH — Attention Exit Message Text

Operation: This is a non-executable module that contains the text for messages issued by IKJEFLG.

Entry from: not applicable

Exit to: not applicable

IKJEFLGM — LOGON Message Handler

Operation: This routine invokes I/O service routines to communicate with either the operator's console or the terminal user during LOGON processing.

Entry from: A LOGON routine.

Exit to: Return to caller.

IKJEFLGN — Message Text for LOGON Message Handler

Operation: This is a non-executable module that contains the text for messages issued by IKJEFLGM.

Entry from: not applicable

Exit to: not applicable

IKJEFLH — LOGON Information Routine

Operation: This routine invokes LISTBC to send MAIL and NOTICES to the terminal user. It issues the __LOGON proceeding__ message at timed intervals until LOGON is complete.

Entry from: IKJEFLC.

Exit to: Return to IKJEFLC.

IKJEFLI — LOGON Installation Exit Interface

Operation: This routine sets up the environment for invoking the installation exit for LOGON (IKJEFLD) and checks the validity of the output from the installation exit.

Entry from: IKJEFLE.

Exit to: Return to IKJEFLE.

IKJEFLJ — LOGON Pre-Attach Exit

Operation: This routine prepares for the ATTACH of the terminal monitor program (TMP). It terminates the LOGON prompting task.

Entry from: IEFSD263 (an initiator routine).

Exit to: Return to IEFSD263.

IKJEFLK — LOGON Post-Attach Exit

Operation: This routine, which is invoked after the terminal monitor program terminates, performs processing for LOGOFF, reLOGON, system-initiated cancellation of the user, or normal cancellation.

Entry from: IEFSD263 (an initiator routine).

Exit to: Return to IEFSD263.

IKJEFLL — LOGOFF Processor

Operation: This routine performs processing for LOGOFF. It updates the UADS data set and analyzes return codes from the job scheduling subroutine (part of the initiator).

Entry from: IKJEFLC.

Exit to: Return to IKJEFLC.

IKJEFLLM — LOGOFF Message Text

Operation: This is a non-executable module that contains the text for messages issued by IKJEFLL.

Entry from: not applicable

Exit to: not applicable

IKJEFLPA — LOGON Time and Date Processor

Operation: This routine obtains the time and data, converts them to text inserts, and puts them into buffers provided by the caller.

Entry from: A LOGON routine.

Exit to: Return to caller.

IKJEFLP0 — LOGON Prompting Monitor Values

Operation: The system generation process creates this non-executable module which contains a LOGON timer value and a LOGON prompting limit.

Entry from: not applicable

Exit to: not applicable

IKJEFLS — LOGON Scheduler's Recovery and Retry

Operation: This routine performs error recording, issues error messages, and schedules a dump, if necessary. Retry processing passes control to an STC module for CSCB clean-up.

Entry from: ABEND processing for the LOGON scheduler task.

Exit to: IEFPRTN for retry or continue with ABEND.

IKJL4T00 — System-Initiated Cancel SRB Routine

Operation: This module synchronizes events between TIOC, LOGON scheduler and its subtasks to provide orderly cancellation of a TSO user.

Entry from: SRB dispatcher.

Exit to: SRB dispatcher.

IKJ5803D — SEND Command Scanner

Operation: This routine scans the SEND command parameter.

Entry from: IEE0403D.

Exit to: IEE0803D, IEE0503D.

ILRACT — Activate VIO ASPCT Routine

Operation: This routine builds a parameter list and calls ILRVSAMI to retrieve the saved VIO ASPCT from SYS1.STGINDEX. The new LGN is stored with the retrieved ASPCT base and the number of slots required to back this VIO data set is recalculated.

Entry from: ILRGOS.

Exit to: ILRGOS.

ILRCMP — I/O Completion

Operation: This module consists of four major routines: the disabled interrupt exit (ILRCMPDI), normal channel end appendage (ILRCMPNE), abnormal channel end appendage (ILRCMPAE), and termination (ILRCMP).

These routines process the individual requests that are represented by the PCCW chained off the IORB. If the I/O is successful the AIAs are returned to page completion and the PCCWs freed. Bad slots will be recorded in a special buffer in SQA.

If the error occurred before the end of the PCCW chain was reached, the remaining PCCWs will be retried.

Entry from:
For entry ILRCMPDI: IECIOSCN.
For entry ILRCMPAE: IECVPST (Post Status).
For entry ILRCMPNE: IECVPST.
For entry ILRCMP: IECVPST or IEAVEDS0, for SRB scheduled by ILRCMP01.

Exit to:
For entry ILRCMPDI: IECIOSCN.
For entry ILRCMPAE: IECVPST.
For entry ILRCMPNE: IECVPST.
For entry ILRCMP: IEAVEDS0.

ILRCMP01 — I/O Completion Recovery Routine

Operation: This recovery routine will attempt to clean up whatever resources have been checkpointed in the ATA and force reprocessing for any requests not yet attempted.

Entry from: IEAVTRTS (RTM).

Exit to: IEAVTRTS.

ILRFMT00 — Format Routine for ASM

Operation: This routine formats ASM and shared RSM control blocks contained in storage areas passed by the system dump-printing routine (AMDPRDMP).

Entry from: AMDPRUIM.

Exit to: Caller.

ILRFMTCV — Common and VIO Areas Format Routine for ASM

Operation: This routine has three entry points. ILRFMTC formats the page and external page tables. ILRFMTH formats RSMHD and its SPCT, ASMHD and its AIA chain, the private area page tables and external page tables. ILRFMTV formats the LGVT and its associated control blocks.

Entry from: ILRFMT00.

Exit to: ILRFMT00.

ILRFMTPG — Paging Format Routine for ASM

Operation: This routine formats the PART, PCTs, PATs, and all other ASM paging-related control blocks.

Entry from: ILRFMT00.

Exit to: ILRFMT00.

ILRFMTSW — Swap Format Routine for ASM

Operation: This routine formats the SART, SPCT, SATs, and all other ASM swap-related control blocks.

Entry from: ILRFMT00.

Exit to: ILRFMT00.

ILRFRR01 — ASM Recovery Service Routines

Operation: This module contains service routines common to ASM recovery routines. There are three types of service routines: 1) queue verification routines that verify and correct ASM queues, 2) control block verification routines that verify ASM-related control blocks, and 3) ASM's resource manager termination routine (parameter for the PURGEDQ macro).

Entries:

1. ILRPSRMT — Reschedule SRB for ILRPTM or ILRSWPDR.
2. ILRVACE — To verify a potential ACE.
3. ILRVACEQ — To verify a queue of ACEs.
4. ILRVACQ2 — To verify and correct a queue of ACEs.
5. ILRVAIA — To verify a potential AIA.
6. ILRVAIAC — To verify a potential AIA/ACE.
7. ILRVAIAQ — To verify a queue of AIAs.
8. ILRVASGQ — To verify ASM staging queue (ASMSTAGQ).
9. ILRVIOE — To verify a potential IOE.
10. ILRVIOEQ — To verify and correct a queue of IOEs.
11. ILRVIORB — To verify a potential IORB-IOSB-SRB chain.
12. ILRVLGE — To verify a potential LGE.
13. ILRVLPRQ — To verify LGE process queue (LGEPROCQ).
14. ILRVPCB — To verify a potential PCB.
15. ILRVPCBQ — To verify queue of PCBs (RSMLIOQ).
16. ILRVPCCW — To verify a potential PCCW.
17. ILRVPCWQ — To verify a queue of PCCWs.
18. ILRVSCCW — To verify a potential SCCW.
19. ILRVSCWQ — To verify a queue of SCCWs.
20. ILRVSPAQ — To verify a queue of swap AIAs.
21. ILRVSWTQ — To verify swap wait queue (SARWAITQ).

Entry from:

For all entries except ILRPSRMT: ILRGOS01, ILRCMP01, ILRSRB01, ILRSRT01, ILRSWP01, ILRTERMR, ILRTMI01, ILRIOFRR.
For ILRPSRMT: IEAVEPDQ.

Exit to:

For all entries except ILRPSRMT: Caller.
For ILRPSRMT: IEAVEPDQ.

ILRFRSLT — Free Slot Routine

Operation: This routine frees slots or swap sets, as required by the caller, by resetting the appropriate PAT bit or SAT bit.

Entries:

ILRFRSLT — used by RSM to free slots.
ILRFRSL1 — used by ASM to free slots.
ILRFRSW1 — used by ASM to free swap sets.

Entry from:

For entry ILRFRSLT: IEAVRELS, IEAVAMSI (RSM routines).
For entry ILRFRSL1: ILRPAGCM, ILRPAGIO, ILRSAV, ILRTERMR, ILRTMRLG, ILRPOS.
For entry ILRFRSW1: ILRPAGCM, ILRSWAP, ILRTERMR.

Exit to: Caller.

ILRGOS — Group Operation Starter

Operation: This module accepts the following group requests: Assign LGN, Save LG/LGN, Activate LG, and Release LG. Each request is attempted to be started immediately. If not started, it is queued for later processing.
Secondary entry (ILRFRELG): Dequeues and frees a LGE.

Entry from:

For entry ILRGOS: Virtual Block Processor (VBP).
For entry ILRFRELG: ILRGOS01 or ILRRLG.

Exit to: Caller.

ILRGOS01 — Group Operators Recovery Routine

Operation: This module serves as an ESTAE for Save and Activate requests and an FRR for Release logical group and Assign requests. It will only retry for record-only abends. For other errors, the resources will be freed and the error percolated to the recovery routine of the caller of ILRGOS.
Entries:
ILRGOS01: Handle Release or Assign errors.
ILRCGOSE: Handle Save or Activate errors.

Entry from:
For entry ILRGOS01: IEAVTRTS (RTM).
For entry ILRCGOSE: IEAVTAS1 (RTM).

Exit to: Caller.

ILRIOFRR — ASM I/O Control Recovery Routine

Operation:
Entry ILRIOFRR: This routine is called by RTM anytime an error occurs during ASM's swap processing, initial page processing, and page completion processing. The mainline of the routine is a router to the appropriate recovery routine or subroutine representing the ASM function in control at the time of the error.
Entry ILRCQIOE: This routine attempts recovery for ILRQIOE, an entry in ILRPAGIO.

Entry from: IEAVTRTS (RTM).

Exit to: IEAVTRTS.

ILRJTERM — ASM Job Termination Resource Manager

Operation:
Entry ILRJTERM: This routine will schedule asynchronous operations to deactivate any VIO data sets still active at job deletion time. The LGE queue is searched to locate these VIO data sets.
Entry ILRJTM01: This routine is the recovery routine for ILRJTERM.

Entry from:
For entry ILRJTERM: IEFSD166 (initiator job deletion module).
For entry ILRJTM01: IEAVTRTS (RTM).

Exit to: Caller.

ILRMSG00 — ASM Message Module

Operation: Entries ILRMSG00 and ILRMSGSP: This routine writes messages concerning the status of page and swap data sets. It will terminate the system if the condition of these data sets make it impossible for ASM to continue.
Entry ILRMSG01: This routine is the recovery routine for the termination subroutine of ILRMSG00.

Entry from:
For entry ILRMSG00: ILRCMP, ILRCMP01, ILRSRT01, ILRPTM, ILRSWPDR.
For entry ILRMSGSP: ILRTMI00 (an ASM initialization routine).
For entry ILRMSG01: IEAVTRTS (RTM).

Exit to:
For entries ILRMSG00 and ILRMSGSP: Caller, if the system is not to terminate. IGFPTERM if the system is to terminate.
For entry ILRMSG01: IEAVTRTS.

ILROPS00 — ASM Page or Swap Open Routine

Operation: This routine is entered to locate, mount, and build I/O control blocks for a page or swap data set. At NIP time control blocks may be requested not to be built. After NIP time contol blocks are always built.

Entry from: ILRASRIM or ILRPGEXP.

Exit to: Caller.

ILRPAGCM — Page Completion

Operation: This routine analyzes an input chain of AIAs, separates them, and places them onto internal queues for the page and VIO subroutine or the swap subroutine. Each subroutine processes its chain of AIAs. Successful AIAs are returned to RSM. Unsuccessful AIAs may be redriven.

Entry from: ILRCMP or ILRPTM.

Exit to: ILRCMP or ILRPTM.

ILRPAGIO — Paging I/O Control

Operation: ILRPAGIO receives a chain of AIA requests from RSM or RSM swap, checks their validity and puts them on the staging queue (ASMSTAGQ). ILRQIOE is then called to build IOEs and drive ILRPTM. Requests for VIO are sent to ILRPOS(VIO Control) prior to processing to determine startability.

Entry from:
For entry ILRPAGIO: ILRSWAP or RSM(IEAVGFA, IEAVSOUT, IEAVOUT, IEAVAMSI, IEAVRFR).
For entry ILRQIOE: ILRPAGIO, ILRSWAP, ILRPOS, or ILRPAGCM.

Exit to: Caller.

ILRPEX — ASM Pool Extender

Operation: This routine attempts to extend the ASM virtual storage pool indicated (ACE, BWK or SWK). ASMVT pool controller information is used and updated.

Entry from: Any ASM module using ILRGMA macro.

Exit to: Caller.

ILRPGEXP — Page and Swap Expansion Module

Operation: This module dynamically adds page and swap data sets to the system when the PAGEADD operator command is issued.
Entry ESTAER is the ESTAE routine for ILRPGEXP.

Entry from:
For entry ILRPGEXP: Attached by IEEVWAIT.
For entry ESTAER: IEAVTAS1 (RTM).

Exit to: Caller.

ILRPOS — Page Operations Starter

Operation:
Entry ILRPOS: This routine receives a string of I/O requests or a single transfer page request. The proper page processing subroutine is called. Requests which cannot be started immediately are queued for later processing.
Entry ILRESTRT: This routine starts AIAs (I/O requests) that are on the LGE process queue.
Entry ILRTRANS: This routine processes transfer page ACEs.
Entry ILRTRPAG: This routine creates a transfer page ACE from the transfer page request (ACA).

Entry from:
For entry ILRPOS: ILRPAGIO or ILRTRPAG.
For entry ILRESTRT: ILRSRBC.
For entry ILRTRANS: ILRSRBC or internally by ILRPOS.
For entry ILRTRPAG: IEAVAMSI (RSM).

Exit to: Caller.

ILRPREAD — ASM Special Read/Write Routine

Operation: This routine is an I/O driver. It builds channel programs to read or write requested slots and invokes IOS using the STARTIO macro to do the I/O. It contains normal and abnormal end appendages and a termination routine. The appendages are null routines. The termination routine posts the mainline when the I/O has completed.

Entry from: For entry ILRPREAD: ILRPGEXP and ASM initialization modules (ILRASRIM, ILRQSRIT, ILRTMI00).
For entries PREADABN and PREADNRM: IECVPST (Post Status).
For entry PREADTRM: IECVPST.
For entry ESTAEXIT: IEAVTAS1.

Exit to: For entry ILRPREAD: Caller.
For entries PREADABN and PREADNRM: IECVPST.
For entry PREADTRM: IEAVEDS0 (Dispatcher).
For entry ESTAEXIT: IEAVTAS1.

ILRPTM — Part Monitor

Operation: The PART contains one entry for each paging data set in use. ILRPTM determines which data sets have page I/O requests queued for them and thus need to have slot sorting performed and I/O initiated.

Entry from: Dispatcher (IEAVEDS0) for SRB scheduled by ILRCMP, ILRCMP01, ILRIOFRR, ILRSRT01, ILRPAGIO, or ILRPTM (itself).

Exit to: IEAVEDS0.

ILRRLG — Release Logical Group Operator

Operation: This routine releases auxiliary storage (unsaved slots) and control blocks (active ASPCTs) relating to a logical group.

Entry from: ILRGOS or ILRSRBC.

Exit to: Caller.

ILRSAV — Save Operator

Operation: This routine stores a logical group's ASPCTs in the SYS1.STGINDEX data set and flags the individual LPMEs (slot information) as saved.

Entry from: ILRGOS.

Exit to: ILRGOS.

ILRSRBC — VIO SRB Controller

Operation:
Entry ILRSRBC: This routine is dispatched in the address space for which a group or page operation is pending. It finds the pending work, determines all work that can be started, and starts it.
Entry ILRSRBRM: This routine cleans up resources. It frees the SRB scheduled to give ILRSRBC control or resets the schedule flag.

Entry from:
For entry ILRSRBC: IEAVEDS0 for SRB scheduled by ILRGOS, ILRVIOCM, or ILRJTERM.
For entry ILRSRBRM: IEAVEPDQ.

Exit to: IEAVEDS0.

ILRSRB01 — Recovery Routine for ILRSRBC

Operation: This routine processes all errors which occur in ASM's SRB controller.

Entry from: IEAVTRTS (RTM).

Exit to: IEAVTRTS.

ILRSRT — ASM Slot Sort

Operation: This routine orders and starts I/O requests which have been queued for a paging data set. This function includes allocating slots within the page data set for write requests.

Entry from: ILRPTM.

Exit to: ILRPTM.

ILRSRT01 — Recovery Routine for ILRPTM and ILRSRT

Operation: This routine eliminates invalid or loop-causing control blocks. The environment is restored or reconstructed such that remaining I/O requests will be properly processed by ILRPTM and ILRSRT.

Entry from: IEAVTRTS (RTM).

Exit to: IEAVTRTS.

ILRSWAP — ASM Swap Control Module

Operation:
Entry ILRSWAP: This routine receives swap requests from RSM and controls the placement or retrieval of the pages on either swap data sets or on page data sets (when no swap data sets are available).
Entry ILRSLSQA: This routine builds and queues SCCWs (swap channel command workareas) for ILRSWPDR to start.

Entry from:
For entry ILRSWAP: IEAVPIOI or IEAVSWIN.
For entry ILRSLSQA: ILRPAGCM or internally from ILRSWAP.

Exit to: Caller.

ILRSWPDR — Swap Driver

Operation: This routine drives IOS via the STARTIO macro for pages being written to or read from a swap data set.

Entry from: IEAVEDS0 for SRB scheduled by ILRSLSQA (entry of ILRSWAP) or ILRCMP.

Exit to: IEAVEDS0.

ILRSWP01 — ASM Swap Recovery Routine

Operation: This routine processes errors which occur in ASM's swapping path. ILRSWP01 entry receives control if there is an error in ILRSWPDR. ILRSCWAP entry receives control if there is an error in ILRSWAP. ILRCSLSQ entry receives control

if there is an error in ILRSLSQA (an entry in ILRSWAP).

Entry from: ILRIOFRR.

Exit to: ILRIOFRR.

ILRTERMR — ASM Address Space Termination Routine

Operation:
Entry ILRTERMR: This routine receives control during termination of any address space. All ASM resources for the address space including storage, control blocks, and auxiliary storage slots are freed or marked to be freed when in-process operations complete.
Entry ILRSLTRV: This routine determines if sufficient unreserved slots exist to allow creation of another address space.
Entry TERMRFRR: Recovery routine for ILRTERMR.

Entry from:
For entry LRTERMR: EAVTMTC (RTM).
For entry ILRSLTRV: IEAVITAS.
For entry TERMFRR: IEAVTRTS (RTM).

Exit to:
For entry ILRTERMR: IEAVTMTC.
For entry ILRSLTRV: IEAVITAS.
For entry TERMFRR: IEAVTRTS.

ILRTMI01 — Task Mode Initialization Recovery

Operation: This ESTAE routine provides recovery for ILRTMRLG and its calls to ILRTMI00 and ILRVSAMI.

Entry from: IEAVTAS1 (RTM).

Exit to: IEAVTAS1.

ILRTMRLG — Task Mode Release Logical Group

Operation: This routine loads and calls ILRTMI00 to complete ASM initialization. It then waits to be posted for Task Mode Release logical group processing (that is, erase saved ASPCTs and release the slots assigned in those ASPCTs).

Entry from: IEAVEDS0 (when initially attached by IEEMB860 or subsequently posted by ILRRLG or ILRTMI01).

Exit to: IEAVEDS0 via WAIT.

ILRVIOCM — VIO Completion

Operation: This routine stores the newly-assigned LSID in an ASPCT for each complete I/O processing of a VIO page.

Entry from: ILRPAGCM.

Exit to: ILRPAGCM.

ILRVSAMI — VSAM Interface Routine

Operation: This routine interfaces with VSAM to complete any I/O to SYS1.STGINDEX involved in processing Save, Activate, and Release logical group operations of ASM.

Entry from: ILRACT, ILRSAV, or ILRTMRLG.

Exit to: Caller.

IRARMCNS — SRM Constants Module

Operation: Provides the pre-assembled tables used within the SRM for non-refreshable data.

Entry from: not applicable

Exit to: not applicable

IRARMCPM — SRM CPU Management

Operation: Consists of a set of routines that monitor the system-wide CPU load. They recommend users for swapping when the system is under- or over-utilized, and users exist that would improve the situation if swapped in or out.

Entry from: IRARMCTL.

Exit to: IRARMCTL.

IRARMCTL — SRM Control Algorithm ALGORITHM

Operation: Routes control to various SRM routines both as a result of explicit requests, and as a result of the expiration of internally scheduled periodic intervals. Defers the routing of control to certain requested routines until concurrently executing SRM processing completes. SRM Control includes the

routines that analyze the system users to determine which should be swapped in or out.

Entry from: IRARMEVT when a SYSEVENT has been received, and from the dispatcher for the execution of an SRM SRB.

Exit to: IRARMEVT or IRARMINT.

IRARMERR — SRM Functional Recovery

Operation: Provides functional error recovery for the SRM.

Entry from: The recovery/termination manager.

Exit to: Recovery/termination manager.

IRARMEVT — SRM SYSEVENT Routers and Processors

Operation: Translates information describing changes in the status of an address space or in the system environment from an external representation to an internal representation that can be acted upon by the SRM. Also performs a limited number of specific services for other OS/VS2 components.

Entry from: IRARMINT.

Exit to: IRARMINT or IRARMCTL.

IRARMINT — SRM Interface Program

Operation: Performs the necessary processing for branch entry and SVC SYSEVENTS to permit information pertaining to individual address spaces or system resources to be passed to and retrieved from the SRM.

Entry from: Any SYSEVENT issuer (branch type SYSEVENT), or SVC processor (SVC type SYSEVENT), or from IRARMEVT or IRARMCTL.

Exit to: SYSEVENT issuer (branch type SYSEVENT), or type 1 SVC exit routine (SVC type SYSEVENT), or to the Dispatcher (if SVC entry and an SRB must be scheduled immediately).

IRARMIOM — and SRM I/O Management

Operation: Consists of a set of routines that monitor the I/O logical channel usage of certain address spaces. These routines recommend address spaces for swapping based upon the extent to which the swap-in or swap-out of the address space would correct a detected I/O system imbalance.

Entry from: IRARMCTL.

Exit to: IRARMCTL.

IRARMIPS — SRM List Processor

Operation: Scans the installation performance specification (IPS) list in the IEAIPSxx member of SYS1.PARMLIB and builds control blocks for IPS information; scans the OPT list in the IEAOPTxx member of SYS1.PARMLIB stores the parameter values in the RMPT.

Entry from: IEEMB812.

Exit to: IEEMB812.

IRARMMSG — SRM Message Module

Operation: Contains the WTO list macro forms of the SRM messages.

Entry from: not applicable

Exit to: not applicable

IRARMRMR — SRM Resource Monitor

Operation: Accumulates several system resource contention indicators, computes the average utilization of these resources and determines if the system multiprogramming level (MPL) should be raised or lowered.

Entry from: IRARMCTL.

Exit to: IRARMCTL.

IRARMSET — SRM Non-Resident Set to New IPS Routine

Operation: Replaces the internal IPS currently in use by the SRM with a new one. Resolves all references in SRM control blocks with offsets or addresses applicable to the current IPS.

Entry from: IRARMEVT.

Exit to: IRARMEVT.

IRARMSRV — SRM Supervisor Service Request Routine

Operation: Requests the invocation of specific supervisor servisor services for SRM routines. The supervisor services requested include reordering the ASCB chain, stealing page frames, and obtaining or freeing storage in the SRM storage pool.

Entry from: An SRM routine.

Exit to: The invoking routine.

IRARMSTM — SRM Storage Management

Operation: Consists of a set of routines that control the use of main storage by all address spaces. For non-swappable users, the mechanism of page stealing is used for storage management control; for swappable users, both page stealing and swapping provide the requisite control.

Entry from: IRARMCTL and IRARMEVT.

Exit to: IRARMCTL and IRARMEVT.

IRARMWAR — SRM Workload Activity Recording Routine

Operation: Collects workload activity data when so requested by MF/1. Terminates the collection of workload activity data when so requested by MF/1, or when the IPS is changed.

Entry from: IRARMEVT or IRARMWLM.

Exit to: IRARMEVT or IRARMWLM.

IRARMWLM — SRM Workload Manager Algorithm Module

Operation: Consists of a set of routines that update user transaction statistics, and monitor the workload level at which individual users are receiving service.

Entry from: IRARMCTL and IRARMEVT.

Exit to: IRARMCTL and IRARMEVT.

IRBMFALL — MF/1 Dynamic Allocation

Operation: Dynamically allocates SYSOUT data sets for MF/1 reports and messages.

Entry from: IRBMFMFC, IRBMFINP, IRBMFRGM and IRBMFSAR.

Exit to: Calling routine.

IRBMFANL — Syntax Analyzer

Operation: Analyzes the data of an input source, as directed by a syntax table. Transfers control to required option initialization routines.

Entry from: IRBMFINP, IRARMIPS, and IRBMFANL (recursive)

Exit to: Caller.

IRBMFCNV — Binary to Character Conversion Routine

Operation: Converts a binary input integer and scale factor to EBCDIC output.

Entry from: IRBMFRGM, IRBMFRCR, IRBMFRPR, IRBMFRWR, IRBMFRDR, IRBMFRHR, IRBMFSAR.

Exit to: Calling routine.

IRBMFDCP — CPU Interval Measurement Gathering Routine

Operation: Calculates CPU wait time during the most recent MF/1 measurement gathering interval. Formats data for SMF record number 70.

Entry from: IGX00014.

Exit to: IGX00014.

IRBMFDDP — Device Interval Measurement Gathering Routine

Operation: Builds the interval image of the SMF device data record for the current interval (SMF record number 74) from data collected in event control blocks by the Device Event Data Collection Routine (IRBMFEDV).

Entry from: IGX00014.

Exit to: IGX00014.

IRBMFDEA — Data Control ESTAE Recovery Routine

Operation: Provides for completion of report formatting of existing measurements when an error occurs.

Entry from: Recovery/Termination Manager (RTM).

Exit to: R/TM.

IRBMFDHP — Channel Interval Measurement Gathering Routine

Operation: Builds the internal image of the SMF channel data record (SMF record 73) for the current interval from data collected by the event driven channel routines IRBMFECH and IRBMFTCH.

Entry from: IGX00014.

Exit to: IGX00014.

IRBMFDPP — Paging Interval Measurement Gathering Routine

Operation: Builds the internal image of the SMF paging data record (SMF record 71) from paging data collected for the current interval.

Entry from: IGX00014.

Exit to: IGX00014.

IRBMFDTA — MF/1 Data Control

Operation: Controls the collection of MF/1 measurement data at specified intervals. Also controls the generation of printed reports after the measurements have been collected.

Entry from: IGX00014 via SYNCH.

Exit to: IGX00014.

IRBMFDWP — Workload Interval Measurement Gathering Routine

Operation: Builds the internal image of the SMF workload record (SMF record 72) at the end of a measurement interval, from data collected by the Workload Manager of the System Resources Manager.

Entry from: IGX00014.

Exit to: IGX00014.

IRBMFECH — Channel Event Data Collection Routine

Operation: Collects channel measurements provided by IOS, and monitors the online/offline status of channels.

Entry from: IRBMFEVT.

Exit to: IRBMFEVT.

IRBMFEDV — Device Event Data Collection Routine

Operation: Collects data concerning the use of I/O devices, as maintained by IOS.

Entry from: IRBMFEVT.

Exit to: IRBMFEVT.

IRBMFEVT — MFROUTER Service Routine

Operation: Calls the event-driven measurement gathering routines (IRBMFECH, IRBMFTCH, and IRBMFEDV) at event indications, according to an input code.

Entry from: System external interruption handler.

Exit to: Caller.

IRBMFFUR — MF/1 Lock Release Functional Recovery Routine

Operation: Called in the event of an MF/1 error to release the dispatcher lock.

Entry from: R/TM.

Exit to: Caller.

IRBMFICP — CPU Measurements Initialization Routine

Operation: Performs initialization necessary so that the interval driven CPU measurement gathering routine (IRBMFDCP) can collect CPU measurements.

Entry from: IGX00013.

Exit to: IGX00013.

IRBMFIDV — Device Measurements Initialization Routine

Operation: Performs initialization necessary so that the interval-driven device measurement gathering routine (IRBMFDDP) can collect device measurements. Also requests activation of the MFROUTER processor to respond to calls for event-driven device measurement gathering.

Entry from: IGX00013.

Exit to: IGX00013.

IRBMFIHA — Channel Measurements Initialization Routine

Operation: Performs initialization necessary so that the interval-driven channel measurements gathering routine (IRBMFDHP) can collect channel measurements. Also requests activation of the MFROUTER processor to respond to calls for event-driven channel measurement gathering.

Entry from: IGX00013.

Exit to: IGX00013.

IRBMFINP — Input Merge Control

Operation: Controls the interpretation and merging of MF/1 control options. Also communicates with the system operator to obtain approval or correction of the controlling option list.

Entry from: IRBMFMFC.

Exit to: IRBMFMFC.

IRBMFIOI — MF/1 IOS Initialization/Termination Routine

Operation: Initiates or terminates collection of device and channel statistics by IOI.

Entry from: IGX00013 and IRBMFTMA.

Exit to: Caller.

IRBMFIPG — Paging Measurements Initialization Routine

Operation: Performs initialization necessary so that the interval-driven paging measurements gathering routine (IRBMFDPP) can collect paging measurements.

Entry from: IGX00013.

Exit to: IGX00013.

IRBMFIWK — Workload Measurements Initialization Routine

Operation: Performs initialization necessary so that the interval-driven workload measurements gathering routine (IRBMFDWP) can collect workload measurements. Also initiates workload

measurement collection by the System Resources Manager.

Entry from: IGX00013.

Exit to: IGX00013.

IRBMFLCV — CPU Report Language Parts Table

Operattion: Provides EBCDIC representations required for the CPU measurements report.

Entry from: not applicable

Exit to: not applicable

IRBMFLDV — Device Report Language Parts Table

Operation: Provides EBCDIC representations required for the Device measurements report.

Entry from: not applicable

Exit to: not applicable

IRBMFLHV — Channel Report Language Parts Table

Operation: Provides EBCDIC representations required for the Channel measurements report.

Entry from: not applicable

Exit to: not applicable

IRBMFLMV — Message Processor Language Parts Table

Operation: Provides EBCDIC representations required for MF/1 messages.

Entry from: not applicable

Exit to: not applicable

IRBMFLPV — Paging Report Language Parts Table

Operation: Provides EBCDIC representations required for the Paging measurements report.

Entry from: not applicable

Exit to: not applicable

IRBMFLTV — Report Header Language Parts Table

Operation: Provides EBCDIC representations for headings required for MF/1 reports.

Entry from: not applicable

Exit to: not applicable

IRBMFLWV — Workload Report Language Parts Table

Operation: Provides EBCDIC representations required for the Workload measurements report.

Entry from: not applicable

Exit to: not applicable

IRBMFMFC — Measurement Facility Control Mainline

Operation: Initializes and controls the operation of MF/1 by connecting to other MF/1 modules.

Entry from: Initiator.

Exit to: Caller.

IRBMFMLN — MFC Mainline Error Analysis

Operation: As the highest level MF/1 ESTAE routine, re-initiates MF/1 after an initial error, or indicates "continue with termination" after a subsequent error.

Entry from: R/TM.

Exit to: R/TM.

IRBMFMPR — MF/1 Message Processor

Operation: Assembles a required message from message parts, and writes out the assembled message line.

Entry from: IRBMFINP, IRBMFRGM, IRBMFMFC, IRBMFDTA, IRBMFSDE, IRBMFMLN.

Exit to: Caller.

IRBMFRCR — CPU Activity Report Generator

Operation: Formats data for the CPU activity report, and writes the data to a SYSOUT data set for either immediate or deferred printing.

Entry from: IRBMFRGM.

Exit to: IRBMFRGM.

IRBMFRDR — Device Activity Report Generator

Operation: Formats data for the device activity report, and writes the data to a SYSOUT data set for either immediate or deferred printing.

Entry from: IRBMFRGM.

Exit to: IRBMFRGM.

IRBMFRGM — Report Generator Control

Operation: Controls the allocation of SYSOUT data space, the calling of report generators for each report type requested, and the freeing of interval measurement data space. Also informs the system operator when reports are ready to print if REALTIME reporting was requested.

Entry from: Initiator.

Exit to: Caller.

IRBMFRHR — Channel Activity Report Generator

Operation: Formats data for the Channel activity report, and writes the data to a SYSOUT data set for either immediate or deferred printing.

Entry from: IRBMFRGM.

Exit to: IRBMFRGM.

IRBMFRPR — Paging Activity Report Generator

Operation: Formats data for the Paging activity report, and writes the data to a SYSOUT data set for either immediate or deferred printing.

Entry from: IRBMFRGM.

Exit to: IRBMFRGM.

IRBMFRWR — Workload Activity Report Generator

Operation: Formats data for the Workload activity report, and writes the data to a SYSOUT data set for either immediate or deferred printing.

Entry from: IRBMFRGM.

Exit to: IRBMFRGM.

IRBMFSAR — Report Generator Control ESTAE Routine

Operation: Attempts recovery from errors in the Report Generator Control routine (IRBMFRGM), or in any of the report generator routines it calls.

Entry from: R/TM.

Exit to: R/TM.

IRBMFSDE — MFSTART ESTAE Recovery Routine

Operation: Controls the removal of all MF/1 supervisor state resources after an error during MFSTART SVC processing.

Entry from: R/TM.

Exit to: R/TM.

IRBMFTCH — Second CPU Test Channel Sampling Routine

Operation: Collects channel busy data (via the test channel (TCH) instruction) at cycle intervals, for MF/1 channel activity data measuring.

Entry from: IRBMFEVT.

Exit to: IRBMFEVT.

IRBMFTMA — MF/1 Termination Mainline

Operation: Disables all MF/1 connections to the resident nucleus during MF/1 termination.

Entry from: IGX00013.

Exit to: IGX00013.

IRBMFTRM — MF/1 General Resource Release

Operation: Removes program and storage resources associated with a single MF/1 measurement function.

Entry from: IRBMFDWP or IRBMFTMA.

Exit to: Caller.

Index

OS/VS2
System Logic Library
Volume 6
SY28-0766-0

READER'S COMMENT FORM

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such requests, please contact your IBM representative or the IBM Branch Office serving your locality.*

Possible topics for comment are:

Clarity Accuracy Completeness Organization Index Figures Examples Legibility

Cut or Fold Along Line

What is your occupation? _____

Number of latest Technical Newsletter (if any) concerning this publication: _____

Please indicate your name and address in the space below if you wish a reply.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

SY28-0766-0

Cut or Fold Along Line

Your comments, please . . .

This manual is part of a library that serves as a reference source for system analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Fold Fold

First Class
Permit 81
Poughkeepsie
New York

Business Reply Mail
No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department D58, Building 706-2
PO Box 390
Poughkeepsie, New York 12602

Fold Fold

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

OS/VS2 System Logic Library Volume 6 (S370-36) Printed in U.S.A. SY28-0766-0