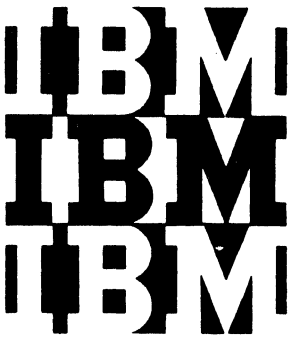


File No. S360-36 (OS)
Order No. GY27-7128-7

Program Logic

IBM System/360 Operating System:

Job Management With MFT,

Program Logic Manual

Program Number 360S-CI-505

OS Release 21

This publication describes the internal logic of the MFT level of job management, its functions, and the control flow among its routines, as MFT job management differs from MVT job management. Included are discussions of system initialization, input stream processing, job initiation and termination, system output processing, command scheduling and execution, and work queue management.

Knowledge of the information in the following publications is required for a full understanding of this manual:

IBM System/360 Operating System: MFT Guide, GC27-6939

IBM System/360 Operating System: MVT Job Management, PLM, GY28-6660

This manual is intended for persons involved in program maintenance, and system programmers who are altering the program design. Program logic information is not necessary for use and operation of the program.

Eighth Edition (March, 1972)

This is a major revision of, and obsoletes, GY27-7128-6 and Technical Newsletter GN28-2468. The text and illustrations have been modified to reflect the changes described in the "Summary of Amendments."

Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

This edition applies to release 21, of the IBM System/360 Operating System, and to all subsequent releases 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/360 and System/370 SRL Newsletter, Order No. GN20-0360, 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, Programming Systems Publications, Department D58, PO Box 390, Poughkeepsie, N. Y. 12602. Comments become the property of IBM.

Preface

The title of this publication, formerly Control Program with MFT, Program Logic Manual, has been changed to Job Management with MFT, Program Logic Manual, to reflect the fact that the document now describes the MFT level of job management only. The information formerly contained in the "Introduction" and the "Initialization of the Operating System" sections of this publication may now be found in the MFT Guide, GC27-6939. The information formerly contained in the "Supervisor" section of this publication may new be found in the MFT Supervisor, PLM, GY27-6736.

This publication describes the internal logic of the MFT level of job management, its functions, and the control flow among its routines, as MFT job management differs from MVT job management. It presents a brief description of each element of job management logic and then indicates if that logic is the same as the corresponding logic in MVT job management. If it is the same, the reader is directed to the publication that contains a detailed description of the MVT logic. Elements of job management logic that are unique to the MFT configuration of the control program are described in detail in this publication.

The manual is divided into five major parts. The "Introduction" briefly describes MFT job management in terms of the various elements used to perform the system initialization, job processing, and command processing functions. It also includes a discussion of job management control flow. Parts 1-4 contain a description of these functions and the common elements of job management. They indicate the areas of logic that are common to MFT and MVT, and describe the processing unique to MFT.

Appendix A contains descriptions of the major tables and work areas used by MFT job management. Appendix B contains descriptions of the modules used by MFT management and includes a table of modules that are unique to MFT, a series of tables listing the modules used by MFT according to major component, and a module cross reference listing. Appendix C contains MFT job management flowcharts. Appendix D contains the acronyms used in the text of the publication.

Readers should have a thorough knowledge of IBM System/360 programming and should be familiar with the basic operation of job management for the MVT configuration of the control program. Knowledge of the information in the following publications is required for a full understanding of this manual:

IBM System/360 Operating System:

MFT Guide, GC27-6939

MVT Job Management, PLM, GY28-6660

This publication makes reference to the following publications:

IBM System/360 Operating System:

IPL/NIP, PLM, GY28-6661

MFT Supervisor, PLM, GY27-7236

Operator's Reference, GC28-6691

System Control Blocks, GC28-6628

Contents

Illustrations

Figures

Charts

Summary of Amendments for GY27-7128-7 OS Release 21

NEW PROGRAMMING FEATURES

Status Display Support
The following task-creating commands have been added to the COMMAND PROCESSING section: DISPLAY PFK, DISPLAY C,K; MONITOR A.

The following existing-task commands have been added to the COMMAND PROCESSING section: MSGRT; STOPMN; CONTROL. The operands in the following list have been moved from the STOP command to the STOPMN command: DSNAME, SPACE, JOBNAMES, STATUS, SESS. Note: For release 21, the system will continue to recognize these operands as valid for use with the STOP command as well as for the STOPMN command.

Console Dump Command
A new SVCLIB module has been added to provide for dumping of main storage to a pre-allocated data set, SYS1.DUMP.

Display SQA Command
A new SVC 34 command has been added to display the system queue area.

Master Scheduler Initialization Routine
Sections of this routine have been rewritten.

Reply Processor for Non-MCS
An SVC 34 routine for non/MCS environments has been added.

Summary of Amendments for GY27-7128-6 as updated by GN28-2468 OS Release 20.1

Name of Item	Description	Area of Publication Affected (Areas Correspond to Entries in Table of Contents)
MODE Command	Modification to SVC 34 routine to provide MODE Command for Model 145.	Appendix B IGF29701
Miscellaneous changes to existing publication	Modifications to SVC 34 and System Task Control routines (routines commonly used by MFT and MVT Control Programs) to provide TSO support.	Appendix B IEEVRC IEFWSMSG IEFWSYP3 Appendix c Charts 10 and 11.

Summary of Amendments for GY27-7128-6 OS Release 20

Name of Item	Description	Area of Publication Affected (Areas Correspond to Entries in the Table of Contents)
MFT Support for Starting Problem Programs from the Console	System task control routines have been modified and an initiator routine (IEFPPGM) has been added to provide the capability of starting any problem program (e.g., TCAM) from the console via a START command.	Part 2 Initiator/Terminator (Scheduler)--Job Selection (IEFSD510) Initiator/Terminator (Scheduler)--Job Selection (IEFSD510): Command Processing Services Initiator/Terminator (Scheduler)--Small Partition Scheduling: Small Partition Module (IEFSD599) Initiator/Terminator (Scheduler)-- Initiator/Terminator Control Flow: Problem Program Initialization Routine (IEFPPGM), Step Initiation Routine (IEFSD512) Part 3 System Task Control (STC) (entire section revised)
CSCB Size Reduction	The size of the CSCB for the MVT configuration of the control program has been reduced. The fields of the MFT CSCB have therefore been reorganized to correspond with those in the MVT CSCB.	Appendix A Command Scheduling Control Block (CSCB)
Separation of a Module of the SVC 34 Routine	Periodic STOP command handler routine IEE4503D has been split into two separate modules. IEE4503D processes periodic STOP commands in a non-MCS environment and IEE5503D processes periodic STOP commands in an MCS environment.	Appendix B Module Descriptions
The MONITOR Command	IEE3503D has been modified to process the MONITOR (DSNAME,JOBNAMES,SPACE, STATUS) commands.	Introduction Command Processing Part 3 SVC 34 Routine

Name of Item	Description	Area of Publication Affected (Areas Correspond to Entries) in the Table of Contents)
Separation of a Module of the System Task Control Routine	Reader control routine IEEVRCTL has been split into two separate modules: reader/interpreter control routine IEEVRCTL and interpreter exit routine IEEVRC.	Part 3 System Task Control (STC)-- START Command Processing: The System Task Control Routines
Background Reader Queue	The queue alter routines, common to both MFT and MVT, have been modified to support the background reader queue, a new subqueue of the job queue. The background reader queue, however, is used only by MVT, not MFT.	Part 3 Master Scheduler Resident Command Processor-- Queue Alter Routine: Queue Alter Delete Routine (IEESD576), Message Routine (IEESD580) Part 4 Work Queues Work Queues-- Queue Management
ASCII Control Program Support	The Job File Control Block and the Step Input/Output Table have been modified to provide the facility for accepting and creating magnetic tapes recorded in ASCII (American National Standard Code for Information Interchange).	Appendix A Job File Control Block (JFCB) Step Input/Output Table (SIOT)
Models 155 and 165 Recovery Management Support	The SVC 34 MODE command processing routines have been modified to provide recovery management support for System/370 Models 155 and 165. IGF2603D is now the router of all of the MODE commands. IGF08501 and IGF08502 replace the former IGF2603D and IGF2703D as the MODE command processors for the System/360 Model 85. IGF29601 processes the MODE command for the Model 155 and IGF55301 processes the MODE command for the Model 165.	Appendix B Module Descriptions
Time-Of-Day Clock	Two new modules: SET Time-Of-Day clock routine IEE6503D and TOD clock TQE update routine IEE6503D have been added to the SVC 34 command processing routines to support the TOD Clock.	Appendix B Module Descriptions

Name of Item	Description	Area of Publication Affected (Areas Correspond to Entries in the Table of Contents)
Delayed Volume Verification	Modifications have been made to the I/O device allocation routines to provide the capability of delaying the verification of volumes with old data sets until the end of allocation.	Appendix B Module Descriptions
Additional changes have also been made to Appendixes A, B, and C to support the items described above. Appendix D: Dictionary of Abbreviations is new for this revision of the publication.		

The primary job management function is to prepare job steps for execution and, when they have been executed, to direct the disposition of data sets created during execution. Prior to step execution, job management:

- Reads control statements from the input job stream
- Places information contained in the statements into a series of tables.
- Analyzes input/output requirements.
- Assigns input/output devices.
- Passes control to the job step.

Following step execution, job management:

- Releases main storage space occupied by the tables.
- Frees input/output devices assigned to the step.
- Disposes of data sets referred to or created during execution.

Job management also performs system initialization functions and the processing required for communication between the operator and the control program. Job management functions may be divided into three major categories: system initialization, job processing and command processing.

System Initialization

The master scheduler task, which performs the system initialization function, is established by the nucleus initialization program (NIP). (See the IPL/NIP PLM) The master scheduler initialization routine receives control after the nucleus initialization program completes the definition of the fixed area of main storage. It passes control to the routines that initialize console communications, the optional system log, the job queue, and the optional system management facility, and define the partitions in the dynamic area of main storage.

Job Processing

Job processing is performed by the reader/interpreter, the initiator/terminator, the system output writer, and direct system output (DSO) processor. The functions of the reader/interpreter are similar to those of the MVT reader; additional information can be found in the MVT Job Management PLM.

After all control statements for a job have been processed, all initiators that are waiting for that job class are posted and the initiator residing in the highest priority partition is given control. The MFT initiator is described in the "Job Processing" section of this publication; for information on allocation and termination, refer to the MVT Job Management PLM.

When the job step has been executed, control is returned to the initiator/terminator which performs data set dispositions and releases input/output (I/O) resources. If the entire job is to be terminated and DSO was not used, the terminator enqueues all data sets on the appropriate system output (SYSOUT) queues.

When the system output writer receives control, it dequeues a job from an output queue, and transcribes the data sets to the user-specified output device. (See the MVT Job Management PLM for further information on the system output writer.)

Command Processing

Command processing is performed by the SVC 34 command scheduler routines, the master scheduler resident command processor routines, and the system task control routines. The SVC 34 command scheduler routines process all commands initially. The job queue manipulation and partition definitions, which are not fully processed by SVC 34, are passed to the master scheduler resident command processor. START commands are processed by the system task control routines. Figure 1 lists the commands used in MFT and indicates the routine which responds to the commands after initial processing.

Command	Responder
CANCEL (active jobs)	Initiator
CANCEL (job in queue)	Master Scheduler
CONTROL K	DIDOCS
DEFINE	Master Scheduler
DISPLAY A,C,K,N,Q,U,jobname, CONSOLES, PFK	Master Scheduler
DISPLAY R	Master Scheduler
DISPLAY SQA	Master Scheduler
DISPLAY T	Timer Maintenance Routine *
DUMP	Master Scheduler
HALT	Master Scheduler
HOLD	Master Scheduler
LOG	System Log *
MODE	Master Scheduler
MODIFY	Task Being Modified
MONITOR A	Master Scheduler
MONITOR STATUS, JOBNAMES, DSNAME	Initiator
MONITOR SPACE	I/O Device Allocation
MOUNT	Master Scheduler
MSGRT (MR)	MSGRT Handler (SVC 34)
RELEASE	Master Scheduler
REPLY	Master Scheduler
RESET	Master Scheduler
SET CLOCK, DATE	Master Scheduler and Timer Maintenance Routine *
SET PROC, Q, AUTO	Master Scheduler
START/STOP	Task Being Started or Stopped
STOPMN (PM)	DIDOCS
SWAP	Master Scheduler
SWITCH	Master Scheduler

Figure 1. Response to Commands After Initial Processing (Part 1 of 2)

Command	Responder
UNLOAD	Initiator
VARY UNIT	Initiator
VARY CH, CPU, PATH, STOR	Master Scheduler
WRITELOG	System Log*
*See the MFT Supervisor PLM, GY27-7236	

Figure 1. Response to Commands After Initial Processing (Part 2 of 2)

Job Management Control Flow

Figure 2 shows the major components of job management and the general flow of control.

Control is passed to job management whenever the supervisor finds that there are no program request blocks in the request block queue. This can occur for two reasons: either the initial program loading (IPL) procedure has just been completed, or a job step has just been executed.

Entry to Job Management After Initial Program Loading

After IPL, certain actions must be taken by the operator before job processing can begin. Therefore, control passes to the communications task which issues a message to the operator instructing him to enter commands, or to redefine the system. If he chooses to redefine the system, control passes to the master scheduler task to handle the redefinitions. If not, the initialization commands (START reader, START writer, and START INIT) are issued either automatically by the master scheduler task or by the operator performing the IPL (see the "Initialization and Restart" section), and job processing begins.

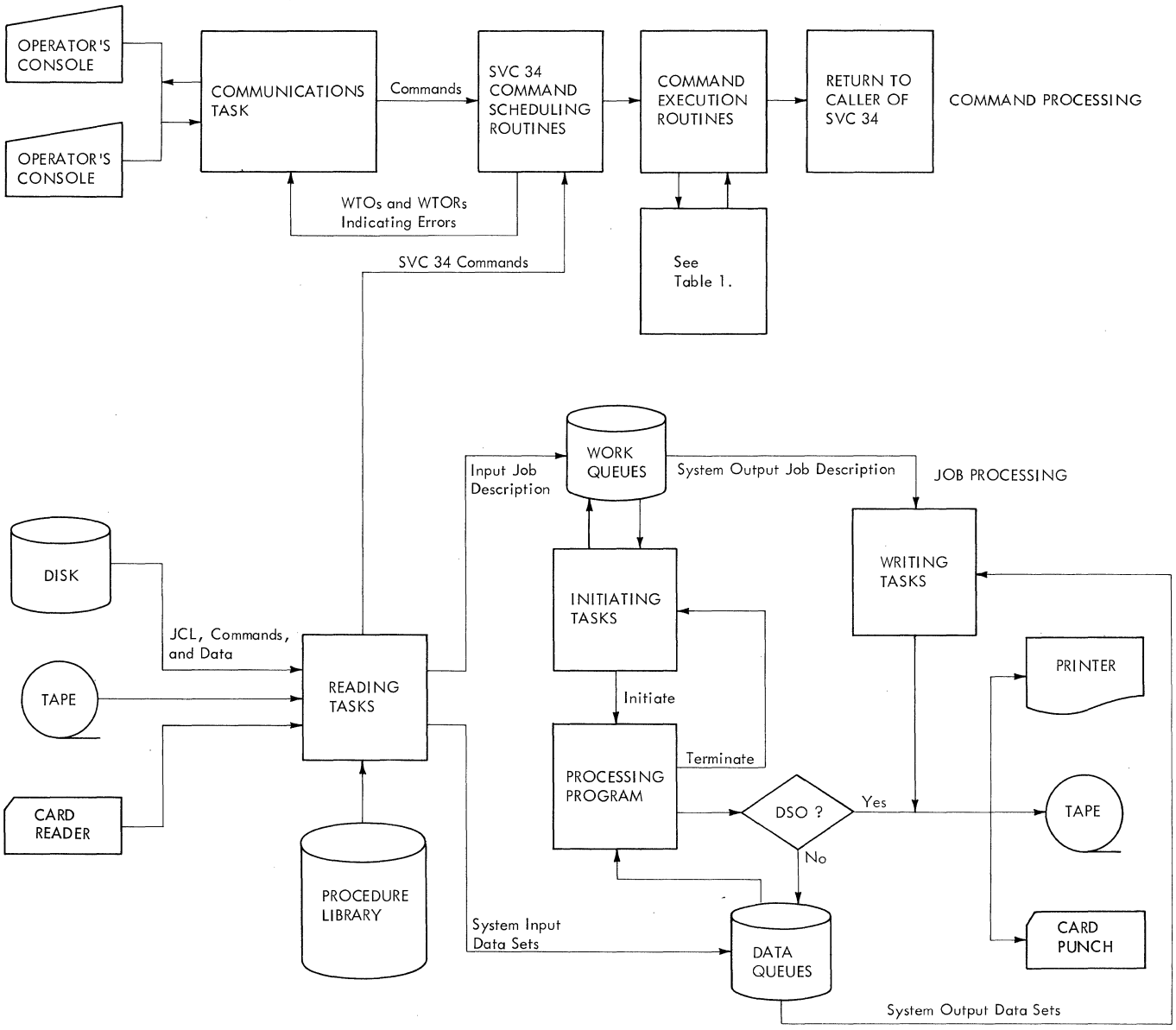


Figure 2. Job Management Data Flow

Entry to Job Management After Step Execution

After step execution, control is passed to the step termination routine of the initiator/terminator. If no further job steps are to be processed, control is also passed to the job termination routine of the initiator/terminator. Both routines are described in the topic "Initiator/Terminator" of the "Job Processing" section in this publication.

PART 1: INITIALIZATION AND RESTART

When the operating system is loaded, it must be initialized to conform to the locations and extents of the system data sets, and to the requirements of the installation. This process, which includes formatting the work queue data set, is called system initialization. If the work queue data set is already in the proper format, special processing must be performed to purge the work queues of incomplete and inappropriate entries; in this case, the processing is called system restart.

System Initialization

There are some major differences in the system initialization processing performed for MFT and MVT configurations of the control program. In MFT, the master scheduler initialization routine (IEFSD569), operating under control of the master scheduler TCB, initializes the dynamic area of main storage. In MVT, this is accomplished by the nucleus initialization program (NIP). In MFT, it is done by the master scheduler to facilitate the redefinition of main storage.

The master scheduler initialization routine (IEFSD569) described below is unique to the MFT configuration of the control program. (For a discussion of the initialization processing performed before the master scheduler initialization routine receives control, see: the IPL/NIP PLM and the MFT Guide.)

The master scheduler initialization routine (Chart 05) first passes control to the communications task initialization routine (IEECVCTI) via a LINK macro instruction. (See the MFT Supervisor PLM.) After the communications task is initialized, the master scheduler initialization routine passes control to the definition routine, IEEDFIN1, via a LINK macro instruction. IEEDFIN1 communicates with the operator, or prepares the partition as it was described at system generation. IEFSD569 then issues the READY message, and if the system log was requested, passes control to IEEVLIN to initialize the system log. (See the MFT Supervisor PLM.) It then types the automatic commands, and issues a WAIT macro instruction.

When the operator presses the REQUEST key, control is given to the supervisor, which recognizes the interruption and passes control to the input/output supervisor. The input/output supervisor determines that the interruption is an attention signal and passes control to communications task console attention interrupt routine. The interrupt routine posts the communications task attention ECB to request reading of the console. The operator enters a SET command. SVC 34 posts the WAIT and places the parameters of the SET command in the master scheduler resident data area. The master scheduler initialization routine then regains control to continue processing. Control blocks for the job queue and procedure library are created. To format the job queue, the routine passes control to queue initialization routine IEFSD055 via a LINK macro instruction which, places a queue control record (QCR) in the nucleus after the DCB and DEB. Control then passes to queue manager formatting routine IEFORMAT, which formats the job queue and returns control to the queue initialization routine. (For a discussion of these two modules, see the topic "Work Queues.") After return from the queue manager initialization routine, the master scheduler initialization module passes control to IEEVPRES for the initialization of volume attributes for all tape and direct access devices. The master scheduler initialization routine then displays and processes any automatic commands.

If the system management facility is specified, the routine stores the SMF options in the first byte of the CVTSMCA field of the CVT. It then passes control via a LINK macro instruction to SMF initialization routine IEESMFIT to initialize the system management facility. (See the "SMF Initialization" topic in the "Common Elements of Job Management" section in this publication.)

The master scheduler initialization routine then establishes partitions based on information in the TCBs. It constructs an RB in each partition, with an XCTL macro instruction addressing job selection module IEFSD510 (for large partitions), or small partition module IEFSD599 (for small partitions). The master scheduler initialization routine then returns control to the dispatcher. The dispatcher returns

control to the master scheduler task, but the TCB now points to master scheduler resident command processor routine IEECIR50 in the nucleus.

System Restart

The system restart functions may be requested at any time that a system restart becomes necessary; e.g., end-of-day, end-of-shift, when a system malfunction occurs, or when power fails. This feature provides a means of preserving a maximum amount of information concerning input work queues, output work queues, and jobs in interpretation, initiation, execution, or termination. System restart permits reinitialization, rather than a complete reformatting, of the job queue data set (SYS1.SYSJOBQE).

MFT uses the MVT system restart modules. For a complete description of these modules, and how they function, see the MVT Job Management PLM.

PART 2: JOB PROCESSING

Job processing is accomplished by three types of tasks:

- Reading tasks, which control the reading of input job streams and the interpreting of control statements in these input streams.

- Initiating tasks, which control the initiating of job steps whose control statements have been read and interpreted. (Terminating procedures are also part of initiating tasks.)

- Writing tasks, which control the transferring of system messages and user data sets from direct-access volumes on which they were written initially to some other external storage medium.

These tasks are created in response to START commands entered for readers, initiators, and writers. Whenever a START reader or writer command is entered, the resulting command processing brings a reader or writer into the associated partition. Initiators are brought into all scheduler-size partitions at system initialization, and after a START INIT command has been issued following partition redefinition. An initiator is also brought into a partition that is specified in a START command issued for a problem program. See the "System Task Control" section of this publication for a description of START commands issued for problem programs.

There may be more than one of each of the job processing tasks so long as the total does not exceed 52. Input job streams may be read simultaneously from three input devices by issuing a START reader command for each input stream. System messages or data sets may be written by system output writers to as many as 36 output devices by issuing a START command for each device. Up to 15 initiating tasks can exist concurrently. Each initiating task is created in response to a START INIT command issued for a specific partition, or a START INIT.ALL command. In addition, each problem program may use direct system output (DSO) processing. DSO is started by entering a START DSO command for a partition naming a system output class and a device. DSO processing is limited only by the number of available devices. (See the Operator's Reference, GC28-6691).

Reader/Interpreter

MFT uses the MVT reader/interpreter (reader). However, because of job class, possible MFT interlocks, and the capability of using transient readers, some modifications have been made to the MVT modules, and six new modules have been added. These modifications and additions are described below.

MFT allows as many as three input readers to execute concurrently with problem programs and writers. Resident readers operate in previously defined reader partitions, and transient readers operate in problem program partitions large enough to accommodate them. Input stream data for the step being read is transcribed onto direct-access storage where it is held until execution of the associated job begins. Problem programs retrieve this data directly from the storage device.

In MFT there are three types of system input readers:

- Resident reader.

- User-assigned transient reader.

- System-assigned transient reader.

Resident and transient readers may operate in the same system, provided no more than one system-assigned reader is specified, and the total number of readers does not exceed three. The primary difference between the user-assigned and system-assigned transient readers is the manner in which the transient reader resumes operation after it is suspended.

RESIDENT READERS

A resident reader operates in a partition designated as such at system generation (by replacing the job class identifier with R), or during system initialization or partition definition (by specifying RDR for the job class identifier). A resident reader reads its input stream, enqueuing jobs until the input stream reaches end-of-file or until it is terminated by a STOP command entered for that partition.

Note: The STOP command does not take effect until the current job is completely read.

TRANSIENT READERS

A transient reader operates in a problem program partition large enough to accommodate it. A transient reader can be terminated by issuing a STOP command or by reaching end-of-file, as can the resident reader. In addition, a transient reader is suspended when a job is enqueued either for the partition occupied by the reader, or for a small partition. (Note that this is possible only when a reader completes reading an entire job.)

If a transient reader is started in a specific partition by including the partition assignment in the START command, it always resumes operation in that same partition, and only when that partition becomes free. This type of transient reader is referred to as user-assigned. If 'S' is substituted for the partition number in the START command, the system assigns the reader to any available large problem program partition. This type of transient reader is called system-assigned.

READER CONTROL FLOW

After a START command is entered to activate a reader, the master scheduler resident command processor routine IEECIR50 determines if the size of the requested partition is large enough, and posts the partition. Job selection routine IEFSD510 determines that a START command has been entered, and passes control to system task control (STC). The STC syntax check routine validates the syntax of the START command, builds job control language tables, and retrieves the reader cataloged procedure specified in the START command. Each reader is assigned to an input device specified in the START command. Control is then passed to interface routine IEFSD533 which sets up an interpreter entrance list (NEL) for a reader. It also allocates job queue space for a transient reader by issuing a dummy WRITE macro instruction. Control is then passed to linkage routine IEFSD537 which issues a LINK macro instruction to reader initialization routine IEFVH1 to begin reading the input job stream (Chart 10).

When reader initialization routine IEFVH1 receives control, it reads its input stream using QSAM, and translates job processing information into convenient form for subsequent processing by an initiator and system output writer. Each job read in by the readers is converted into tables that are placed in the appropriate job class input work queue specified by the CLASS parameter on the JOB statement. One input work queue exists for each of the fifteen problem program job classes (A through O).

For systems that include Multiple Console Support (MCS), the PARM field on an EXEC statement includes a command authority code. This code is included in the option list created by interface routine IEFSD533, and placed in the interpreter work area (IWA) by reader initialization routine IEFVH1. This code is passed by the reader when it issues an SVC 34 due to a command read in the input stream.

After the reader has completed reading a job, control passes to queue manager enqueue routine IEFQMNQQ which enqueues the job on the appropriate input work queue according to the PRTY parameter on the JOB statement (see "Queue Management" in this section).

Note: If the reader is being used as a subroutine by a problem program, it does not enqueue the job on the input work queue, but returns control to the problem program passing the addresses of the JCT constructed for that job, and the QMPA associated with that input queue entry.

If data is encountered in the input stream, control is passed to interpreter CPO routine IEFVHG to transcribe the data onto direct-access storage for later retrieval by the problem program. If there is no space for the data, control passes to interpreter operator message routine IEFSD536 to issue a DISPLAY active command and a WTOR message. The operator replies with either 'WAIT' or 'CANCEL'. If 'WAIT' is specified, the reader waits for space to become available. If 'CANCEL' is specified, the reader is canceled and a READER CLOSED message is issued. IEFSD536 then sets indicators which cause cleanup of the current job, and control to be passed to interpreter termination routine IEFVHN to terminate the reader.

After a reader enqueues each job, control passes to transient-reader suspend tests routine IEFSD532. This routine decides whether to 1) terminate the reader, 2) suspend the reader, or 3) have the reader continue reading the job stream. (The decision to suspend the reader would never be made if the reader is resident.) If the reader is to be terminated, control passes to termination routine IEFVHN. If the reader is to be suspended, control passes to transient reader suspend routine IEFSD530. Otherwise, control returns to job and step enqueue routine IEFVHH to continue reading the job stream.

Transient Reader Suspend Routine (IEFSD530)

When a transient reader is suspended, transient reader suspend routine IEESD530 (Chart 13) writes the tables and work areas used by the reader onto the work queue data set (SYS1.SYSJOBQE).

The routine closes the reader and procedure library. Data needed to restore the reader is temporarily saved in the interpreter work area (IWA). The IWA is then written to the work queue data set. When a user-assigned transient reader is suspended, the address of the reader space on the work queue is placed in the partition information block (PIB). When a system-assigned transient reader is suspended, the address of the IWA is placed in the master scheduler resident data area (IEESD568). (See Appendix A for the format of the master scheduler resident data area.) The work queue data set is later used by transient reader restore routine IEESD531 to restore the reader when the assigned partition becomes available after job termination. "No work" ECBs for problem program partitions are posted (see "Job Selection"), and the JCTJMR field of the JCT is tested to determine if SMF is supported. If this field contains zeroes, there is no job management record (JMR) and SMF is not supported. If SMF is supported, the user's SMF exit routine IEFUJV (whose address is contained in the JMRUJVP field of the JMR) is deleted if it is present in main storage. Storage for the JMR is also freed via the FREEMAIN macro instruction.

The transient reader suspend routine then returns control to system task control.

Note: See the MVT Job Management PLM for the format and description of the JMR.

Transient Reader Restore Routine (IEFSD531)

Once a partition is again free for the reader, transient reader restore routine IEFSD531 (Chart 14) receives control and issues a GETMAIN for the IWA, Local Work Area (LWA), reader DCB, and procedure library DCB. The direct-access device address of the IWA is retrieved from the PIB if a user-assigned reader is to be restored, or from the master scheduler resident data area, if a system-assigned reader is to be restored. The IWA is then read in from the job queue. The TIOT is read into storage and the TCB pointer is updated; other tables and work areas necessary to restore the reader are reset from the information saved in the IWA.

If SMF is in the system and if SMF options are specified, a GETMAIN macro instruction is issued to obtain main storage for the job management record (JMR). The JMR is then initialized with the SMF options and the RDR device type and name. If SMF exits are specified, the name of SMF user exit routine IEFUJV is placed in the interpreter entrance list (NEL). The routine is then loaded and its address is placed in the JMRUJVP field of the JMR. The reader and procedure library DCBs are opened and the reader resumes operation to start reading at the point in the job stream where it was suspended. Control is then passed to interpreter routine IEFVHCB to continue reading the job stream.

Initiator/Terminator (Scheduler)

To provide independent scheduling, schedulers operate in any problem program partition of sufficient size. A partition large enough to accommodate the scheduler is referred to as a "large partition." A partition not large enough to accommodate the scheduler is referred to as a "small partition". Within a given large partition, a scheduler operates independently of schedulers in other large partitions. Because small partitions cannot accommodate the scheduler, they rely on large partitions to perform their initiation, allocation, and termination operations. Scheduling for small partitions is described in "Small Partition Scheduling" in this section.

An MFT initiator (Chart 8) dequeues a job (entry) for its partition based on a job class designated for the partition. Once dequeued, the job is scheduled according to the information contained in the entry.

During allocation and termination of each job step, the allocation and termination routines place messages and output data set pointer blocks in a specified output queue. The queue entry is created by the reader/interpreter. (The output queue entry becomes input to an output writer when the job is completed.)

An initiator functions as a control program for the scheduling process, using the allocation and termination functions as closed subroutines. (See Figure 3 for an illustration of the scheduling process in a large partition.) The MFT initiator is composed of the following routines:

- Job Selection
- Small Partition
- Job Initiation
- Data Set Integrity

- Step Initiation
- Problem Program Interface
- Step Deletion
- ENQ/DEQ Purge Routine
- Alternate Step Deletion
- Job Deletion

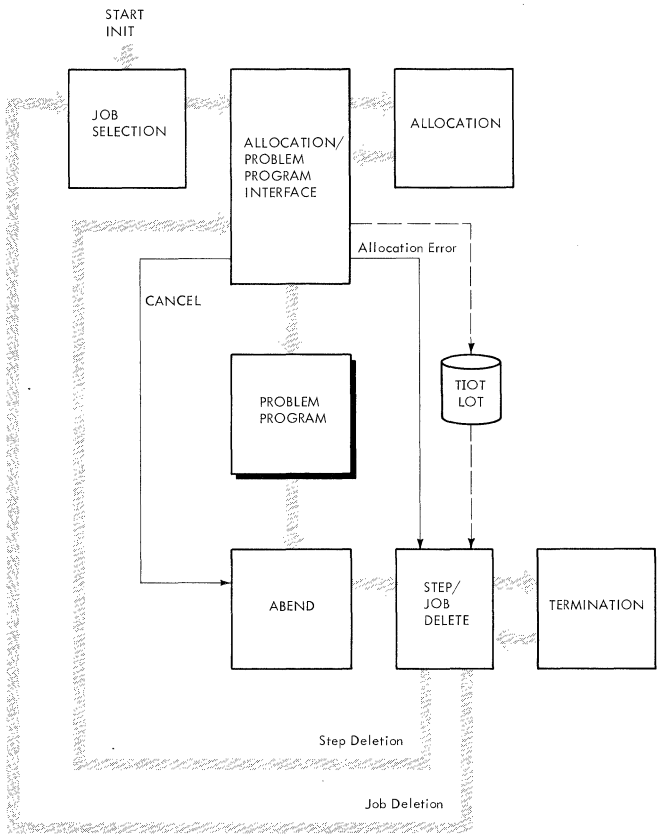


Figure 3. Scheduling a Problem Program Entered Through the Input Stream in a Large Partition

JOB SELECTION (IEFSD510)

The job selection routine (Chart 9)acts as the control routine for the MFT initiator. The routine is brought into all large problem program partitions by the master scheduler at system initialization, by the job deletion routine when a job has terminated, or by system task control when a system task has been scheduled for a small partition or a system task has been suspended.

Job selection first waits on a "no work" ECB in the PIB. This ECB is posted complete by the command processing routines, the job deletion routine, system task control, or the small partition module when a small partition needs scheduler services.

When the "no work" ECB has been posted complete, the job selection routine checks the PIB to determine if a life-of-task (LOT) block exists (see Appendix A for a description of the LOT block). If not, it creates one for the task.

Job selection then checks the PIB for a small partition information list (SPIL) pointer (see Appendix A for a description of SPIL). If one exists, scheduling is performed for the small partition by passing control to IEFSD599. If no SPIL pointer exists, the PIB is checked for any pending STOP DSO or MODIFY DSO commands. These are processed by passing control to stop and modify command processing routine IEFDSOSM.

Upon return from IEFDSOSM, the PIB is checked to determine if the partition is involved in partition redefinition; if the partition is to be changed, the PIB is checked further. If a job is queued on the checkpoint/restart internal queue it is processed; if a restart reader is pending, it is started. If neither exists, any DSO processing is stopped, no further scheduling is allowed in the partition and the partition can be redefined. (See "The Master Scheduler Resident Command Processor.")

If the partition in which the initiator is operating is not part of a partition redefinition, a test is made for a pending Restart Reader command. If no command is pending, a test is made to determine if a system task is to be started. If a restart reader or a system task is to be started, control passes to system task control. If a restart reader is being started, and a user-assigned reader had been rolled out of the partition, the PIB is marked accordingly.

If no small partition is requesting service, no system task is to be started, and the partition is not part of a redefinition operation, a final check is made to determine if a START INIT command has been issued; if so, job selection attempts to dequeue work from the input work queue. If a STOP INIT command has been issued, the attempt to dequeue a job is bypassed.

A threshold check is then made to determine if enough logical tracks are available on SYS1.SYSJOBQE to start the initiator. If not, message IEF427I COMD REJECTED FOR INITIATOR 'ident' - INSUFFICIENT QUEUE SPACE is sent to the operator and job selection again waits on the "no work" ECB.

The job selection routine obtains storage for the job control table (JCT)

andchecks to determine if a job is queued on the checkpoint/restart internal queue. If a job exists, dequeue by jobname routine (IEFLOCDQ) is used to remove it from the hold queue for processing. If no job is on the internal queue, the routine then uses the queue manager dequeue routine (IEFQMDQQ) to obtain work from one of the input job queues according to the job class assignment of the partition. If work is found, IEFQMDQQ constructs a CSCB for the job and an IOB to be used when reading or writing the input queue. The CSCB is constructed in the system queue area and the address of the CSCB is placed in the LCT. The address of the IOB is placed in the queue manager parameter area (QMPA). When a user accounting routine is supplied, the job selection routine sets all four fields of the timer work area in the LCT to zero. These fields are used in calculating the execution time of a job step. Job selection then branches to job initiation routine IEFSD511.

If the search for work for the partition is unsuccessful (i.e., no work has been enqueued for any of the job classes assigned to the partition) tests are made to determine if a transient reader is to be restored in the partition or if a START command has been entered for a system-assigned transient reader. If so, system task control is called. If a reader is to be restored in the partition, job selection passes control to system task control linkage routine IEFSD588.

Command Processing Services

In response to commands entered in the input stream or from a console, the command processing routines request a service by storing information in the PIB of the affected partition or in the master scheduler resident data area for START and STOP commands issued for system-assigned transient readers and writers. The job selection routine recognizes these requests and takes one of the following actions:

- Inhibits further job scheduling for the partition in preparation for the processing of a DEFINE command. (The DEFINE command can be entered only from a console.)
- Prevents execution of problem programs in large partitions in response to either a STOP INIT command specifying a particular partition or a STOP INIT.ALL command.
- Passes control to system task control (STC) in response any START command other than a START INIT command.
- Schedules problem program execution in response to either a START INIT command or a START command issued for a problem program in that partition.

SMALL PARTITION SCHEDULING

A partition is defined as "small" when its size is at least 8K bytes but less than the job scheduler generated for the system. Small partition scheduling is performed by an initiator in a scheduler-size partition at the request of small partition module IEFSD599 (IEFSD599 is described later in the topic "Small Partition Module"). The small partition is therefore temporarily dependent on a large partition while scheduler services are being performed. Scheduling for a small partition is independent of scheduling for other small partitions in the system.

The small partition module interfaces with job selection module IEFSD510 to schedule a problem program or to establish an interface with system task control for the scheduling of a system task in a small partition. Communication between the small partition module and IEFSD510 or system task control is maintained through a small partition information list (SPIL). (The format of a SPIL is shown in Appendix A.)

Small partition module IEFSD599 requests the scheduling function by placing the address of a SPIL in the partition information block (PIB) of each scheduler-size partition in the system. Each time that job selection module IEFSD510 is entered between jobs, it checks the PIB for a nonzero SPIL address. If the PIB contains a valid SPIL address, IEFSD510 performs one of the following services for the small partition:

- It tests to determine if there is work for the small partition and if so, IEFSD510 passes control to job initiation routine IEFSD511. If not, it notifies the small partition accordingly.

- If the small partition is waiting for termination services, IEFSD510 passes control to step deletion routine IEFSD515.

- If a START command has been entered on the console specifying a system task or a problem program in a small partition, IEFSD510 passes control to system task control.

These routines perform the requested service in the large partition and use the SPIL to indicate their action to IEFSD599. When the requested service has been performed, these routines return to IEFSD510.

Initiating a Problem Program

As shown in Figure 4, initiation of a problem program in a small partition is performed by a large partition. If a small partition is waiting for work, job selection module IEFSD510 dequeues a job from an input work queue that the small partition is assigned to service. The large partition posts a completion code in field ECBA of the SPIL when initiation services have been performed.

A completion code of one indicates that no work was found for the small partition. The small partition then waits on the ECB list in the SPIL. The posting of any of the listed ECBs causes the small partition to request initiation services.

A completion code of zero indicates that initiation services have been performed and the problem program job step is ready to be executed. The small partition, using the allocate parameter list (APL), moves the task input/output table (TIOT) and life-of-task (LOT) block from the large partition, opens required DCBs, and establishes problem program mode. (If the system has the storage protection feature, the protection key is set.) If the job has not been canceled, control passes to the problem program, thus freeing the large partition to continue processing.

Initiating a System Task

As shown in Figure 5, if a system task is to be started in the small partition, small partition module IEFSD599 requests the services of job selection module IEFSD510 for initiation of the system task. IEFSD510 responds to the request by bringing the first system task control (STC) module into the large partition. STC performs the initiation functions up to the point of passing control to the system task. STC write TIOT routine IEFSD590 then posts the ECBA in the SPIL with a completion code of zero to indicate to IEFSD599 that initiation services have been performed, and that the system task is ready to be executed. The small partition module then uses the link parameter list

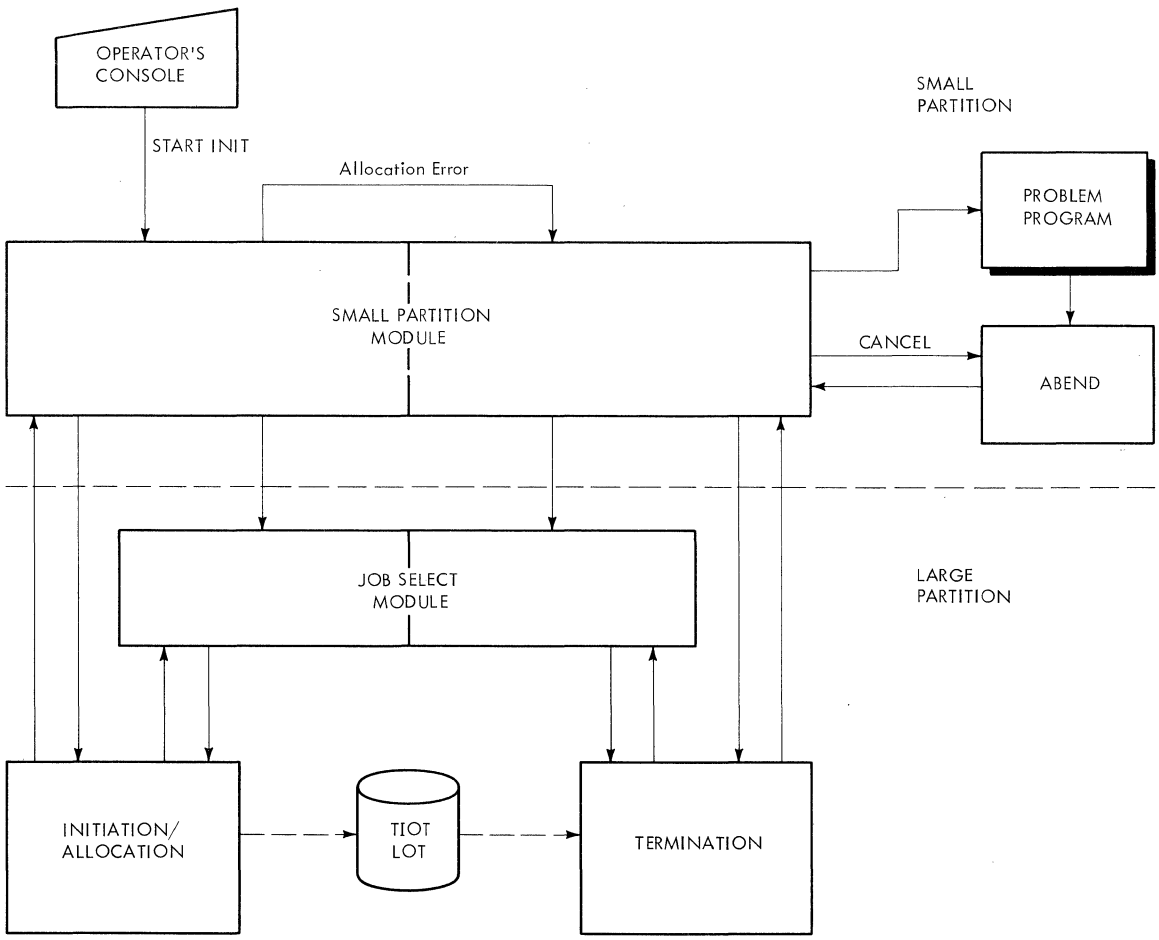


Figure 4. Scheduling a Problem Program Entered Through the Input Stream in a Small Partition

(LPL) to move the TIOT from the large partition to the small partition. It posts the ECBC in the SPIL, thus freeing the large partition to continue normal processing. IEFSD599 then frees the main storage occupied by the SPIL and passes control to the system task via linkage modules IEF589SP, IEFSD584, and IEFSD591. (See Chart 17).

Terminating the Small Partition

When the job step is completed, or a writer is stopped, small partition module IEFSD599 is brought back into the partition and entered at special entry point SMALLGO. A check is made to determine whether a scheduler ABEND occurred. If it did, a message is issued to the operator with a completion code, and all CSCBs associated with that job are removed from the CSCB chain. Control then passes to the normal entry point of IEFSD599. If no scheduler ABEND occurred, IEFSD599 determines if job step timing is being performed by testing the high-order bit in the job step timing status bits field of the PIB. If the bit is on, the TQE is being used for job step timing and the routine issues a TTIMER macro instruction to stop the timing and to obtain the step time remaining for use in updating the SPIL. It then turns off the bit and saves the step time remaining in a register until the SPIL is created. When the SPIL is created, the routine updates it with the step time remaining and sets the status bit indicating that termination services are requested. The small partition module then begins a search for a large partition to perform the job termination required.

After an initiator in a large partition has performed the termination services, ECBA in the SPIL is posted with a completion code of two to indicate that job termination has taken place. A check is made to determine if the small partition is involved in a redefinition operation. If it is, the small partition is made quiescent. If the small partition is not associated with a redefinition operation, it requests additional services from an initiator in a large partition.

Note: If the initiator in a large partition performs step termination instead of job termination, the next step of the job in the small partition is scheduled before the initiator schedules a job into its partition, or before it performs scheduling services for another small partition.

Small Partition Module (IEFSD599)

Small partition module IEFSD599 (Chart 1) is entered from the redefinition routines at system initialization or when a DEFINE command is issued or from the master scheduler. The module is entered at special entry point SMALLGO from the ABEND routines when a step has completed execution. IEFSD599 first waits on a "no work" ECB located in the partition's PIB. When this ECB is posted complete, the PIB is checked to determine if a SPIL has been created. If not, one is created and an indicator is set in the PIB. The PIB is then checked for pending STOP DSO or MODIFY DSO commands. IEFSD599 passes control to stop and modify command processing routine IEFDSOSM to process any such pending DSO commands.

Upon return from IEFDSOSM, IEFSD599 checks the PIB to determine if the partition is involved in a redefinition operation. If a redefinition is pending, the internal job queue of checkpoint/restart jobs is checked and any jobs on the queue are processed before the partition redefinition. If there is nothing on the internal job queue and redefinition is pending, assigned tracks are deleted, the SPIL is freed, any DSO processing is stopped, and pending CSCBs are freed. The 'DEFINE' ECB in the PIB is posted to indicate that the partition has been made quiescent, and a return is made to wait on the "no work" ECB.

If no redefinition operation is pending, the PIB is checked to determine if a system task is to be started in the partition. If so, an indicator is set in the SPIL, assigned tracks are deleted, and a request for scheduling is made to a large partition (described below). If a system task is not to be started, the STOP INIT bit in the PIB is checked. If this bit is on, assigned tracks are deleted, the SPIL is freed, and a return is made to wait on the 'no work' ECB. If the STOP INIT bit is not on, the PIB is checked for track assignment. If needed, tracks are assigned and indicated in the PIB. The SPIL is updated to indicate a request for initiation of a problem program.

A request is made for a large partition to service the small partition based on the contents of the SPIL. First, an exclusive ENQ macro instruction is issued to prevent concurrent service requests by small partitions. Interruptions are disabled to prevent interference with the address of the SPIL in the large partition's PIB. IEFSD599 then searches for a scheduler-size partition. The TCBs are tested for problem program status; when a scheduler-size partition is found, a determination is made of whether the small partition is involved in a DEFINE operation.

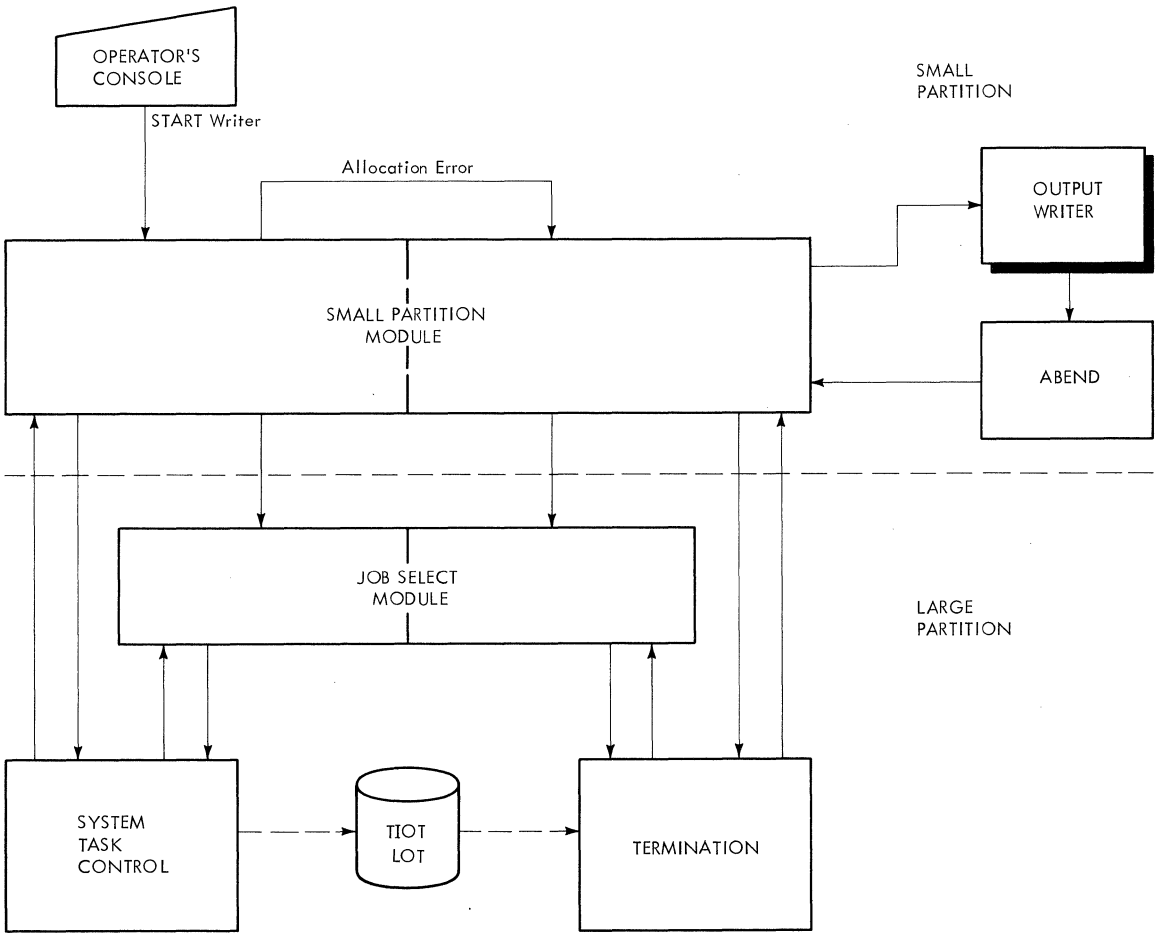


Figure 5. Scheduling a System Task in a Small Partition

If the small partition is involved in a DEFINE operation, the test for the large partition involved in a DEFINE operation is bypassed. If the small partition is not involved in a DEFINE operation, the large partition is tested to determine if it is involved in a DEFINE operation. If so, the large partition is bypassed and the TCB search is continued.

The address of the SPIL is stored in the PIB of the large partition, thus constituting a request. An indication is made when storing occurs. If a large partition is waiting on its 'no work' ECB (in its PIB), the large partition is posted and the large partition routine clears the SPIL addresses in the other large partition PIBs. When a large partition is posted, or all applicable TCBs are checked, interruptions are enabled.

If no SPIL pointers were stored during the search, a DEQ macro instruction is issued (to allow other small partitions to make requests), and a WAIT macro instruction is issued on a 'dormant' ECB in the small partition's PIB. (When later posted by the command processing routines, the small partition module will repeat its search). If at least one SPIL pointer was stored, a WAIT macro instruction is issued on ECBB in the SPIL. This allows a large partition, immediately upon recognition of the request, to post the ECB complete. The small partition module may then issue a DEQ macro instruction to release the SPIL pointer field so other small partitions may make requests.

Next, a WAIT macro instruction is issued on ECBA (in the SPIL) to delay the small partition until the requested service has been performed. When ECBA is posted complete by the large partition, the completion code is tested to determine the action which occurred. If the completion code is two, job termination occurred and return is made to the point of determining the DEFINE status of the small partition. If the completion code is one, 'no work' was found for the small partition and a return is made to WAIT on the ECB list in the SPIL. If the completion code is zero,

the large partition is at the point of calling either the problem program or a system task. The large partition is waiting on ECBC (in the SPIL) to allow transfer of information into the small partition by the small partition module.

If a problem program is to be initiated, IEFSD599 uses the allocate parameter list (APL) to move the TIOT and user parameter area into the small partition. It then posts ECBC (freeing the large partition), and opens Fetch and/or JOBLIB DCBs if required. To process write-to-programmer messages during problem program execution, IEFSD599 puts the address of the SYSOUT QMPA into the WTPCB, which is located in the CSCB.

The routine then determines if job step timing will be performed by testing the step time limit in the timer work area of the LOT block. If this value is equal to 24 hours, the job step will not be timed. If the job step is to be timed, IEFSD599 issues the STIMER macro instruction to set up the step time interval. The routine then sets bit zero of the job step timing status bits field of the PIB to one indicate that the job step TQE is being used by the Initiator. It also sets bit one to one to indicate to step deletion routine IEFSDS15 that the STIMER macro instruction was issued specifying the TQE addressed in the PIB.

The small partition routine establishes the partition in the problem program protection mode and frees the SPIL. If the program to be initiated is the DSDR processing step of a checkpoint restart, IEFSD599 uses the APL to move the TIOT and user parameter area into the small partition, and posts ECBC. The routine moves the job QMPA and the SYSOUT QMPA from the LOT to the CSCB, and bypasses opening the JOBLIB and FETCH DCBs. The routine also bypasses setting the storage protection key but frees the SPIL.

A check is made to determine if the job has been canceled. If so, an ABEND macro instruction is issued. If the job has not been canceled, an XCTL macro instruction is issued to call the problem program into the small partition (the problem program passes control to ABEND at completion of its execution).

ABEND recalls the small partition routine and enters at special entry point SMALLGO. The routine changes the small partition protection key to zero. If job step timing is being performed, it issues the TTIMER macro instruction to stop the timing and to obtain the step time remaining for use in updating the SPIL. It sets bit zero of the job step timing status bits field in the PIB to zero to indicate that the job step TQE is no longer active. After it it creates the SPIL, the routine updates it with the step time remaining and turns on the status bit indicating that termination services are requested. IEFSD599 then begins the search for a large partition to service the request.

INITIATOR/TERMINATOR CONTROL FLOW

There are no terminator routines that are unique to MFT; the modules used in MFT task termination are described in the MVT Job Management PLM.

In addition to IEFSD510 and IEFSD599, several other initiator routines are unique to MFT. These are described in the following paragraphs. Descriptions of the MVT allocation and step initiation routines that have not been modified by MFT can be found in the MVT Job Management PLM.

Problem Program Initialization Routine (IEFPPGM)

Problem program initialization routine IEFPPGM receives control from system task control linkage routine IEFSD589 for START commands issued for problem programs. It obtains main storage for a LOT block, if one does not exist, and initializes it. It also reinitializes fields in the CSCB, the PIB, and, for small partitions, the SPIL.

The routine then tests the CHSPA field of the CSCB to determine if any internal JCL or I/O errors occurred during reader/interpreter processing. If either of these types of errors occurred, the problem program initialization routine cancels the starting task and issues an appropriate message to the operator.

IEFPPGM then makes a threshold check to determine if enough logical tracks are available on SYS1.SYSJOBQE to start an initiator. If not, it cancels the job and issues a message to the operator indicating this action.

Finally, the problem program initialization routine passes control to job initiation routine IEFSD511 via an XCTL macro instruction.

Job Initiation Routine (IEFSD511)

Job initiation routine IEFSD511 issues a GETMAIN specifying subpool 0 to obtain space for the system output class directory (SCD). The SCD is then read into the area and the contents of the SCD are used to initialize QMGR2 in the LOT block. (QMGR2 is the queue manager parameter area which is used for referencing the output data

set.) After QMGR2 has been initialized, the storage obtained for the SCD is freed. A GETMAIN is then issued to obtain storage for IOB2, the IOB used in conjunction with QMGR2. A GETMAIN is issued (specifying subpool 253) to obtain space for the step control table (SCT). The SCT is read into the area thus obtained. Job initiation then branches to data set integrity routine IEFSD541.

If direct system output (DSO) processing is available in the partition, job initiation uses the SCD to build a table of all classes of SYSOUT, including the message class, contained in the job stream. Job initiation uses this table to determine if DSO is available for the job; if so, it selects DSOCBs for the job. Selection of a DSOCB is indicated by placing the problem program's protection key into the DSOCB and flagging the job's JCT.

Data Set Integrity Routine (IEFSD541)

The data set integrity routine is entered only once per job, from job initiation routine IEFSD511. It first determines whether data set integrity processing is required.

If the JCT indicates a 'failed' job or if there are no explicit data sets (DSNAME parameter in a DD statement) for the job, processing is bypassed and exit is made to step initiation routine IEFSD512. If data set integrity processing is required, the DSENQ table records are read from the job's entry in the input job queue (SYS1.SYSJOBQE). Duplicate DSNAMEs are eliminated from the table and each unique DSNAME is placed in a minor name list. The most restrictive attribute (exclusive or share) is chosen for each DSNAME placed in the minor name list. After this processing is complete, an ENQ supervisor list is constructed which contains an entry for each DSNAME in the minor name list. Each entry is initialized with the following:

- RET=TEST option of ENQ.
- SYSTEM option of ENQ.
- Attribute (E/S) of the corresponding DSNAME.
- Address of the common major name 'SYSDSN'.
- Address of the corresponding DSNAME (considered the minor name) in the minor name list.

The DSNAME (minor name) length is contained in the first byte of each DSNAME field in the minor name list.

When the ENQ supervisor list is constructed, the system is disabled and an ENQ supervisor call is issued against the list to test the availability of the DSNAMEs. If the DSNAMEs are available, the ENQ supervisor list is updated so that each entry reflects the RET=NONE option of ENQ. A second ENQ supervisor call is issued against the list to reserve DSNAMEs for the job. The system is enabled and exit is made to step initiation routine IEFSD512.

If the DSNAMEs are unavailable for the job (already reserved with conflicting attributes by other task(s) in the system), the operator is notified of the condition. In notifying the operator, the return code field of each entry in the ENQ supervisor list is tested for a nonzero setting. If the setting is nonzero, the associated DSNAME (minor name) is identified to the operator as unavailable. The operator is given the following reply options:

- RETRY, in case the resources have been freed by the other task(s) (processing is delayed until the operator replies).
- CANCEL the job.

If RETRY is entered by the operator, processing continues at the initial ENQ supervisor call to again test the availability of the DSNAMEs. The operator is again notified, and he can reply either RETRY or CANCEL. If the job is canceled by the operator, the 'job fail' bit in the JCT is set and exit is made to step initiation routine IEFSD512.

Step Initiation Routine (IEFSD512)

Step initiation routine IEFSD512 first issues a GETMAIN macro instruction to obtain storage for a 72-byte register save area for SMF user initiation exit routine IEFSMFIE and branches to IEFSMFIE. Upon return, it frees the register save area and tests to determine if job step timing will be performed. If the job time limit in the JCT is equal to 24 hours, the job step will not be timed. In this case the step initiation routine moves the 24 hour limit to the timer work area in the life-of-task (LOT) block, and bypasses the procedure for setting up the step time limit.

If the job time limit in the JCT is equal to any value other than 24 hours, IEFSD512 determines the value to be used as the step time limit in the timer work area of the LOT block. For each step of the job, the routine determines if allowing the step to use the full amount of time specified for it would cause the job time limit to be exceeded: IEFSD512 calculates the amount of job time remaining by subtracting the job time used from the job time limit and compares this figure with the step time limit. It establishes the step time limit by placing the smaller of the two figures in the step time limit field of the timer work area. If the

smaller of the two figures is the job time remaining, the routine turns on the high order bit in the step time remaining field of the timer work area to indicate that the job time remaining is being used as the step time limit.

IEFSD512 then issues a GETMAIN specifying subpool 253 to obtain storage for an allocate register save area (ARSA) and an allocate parameter list (APL). The APL (Figure 6) is initialized containing addresses of the LOT, JCT, and SCT, and two words of zeros.

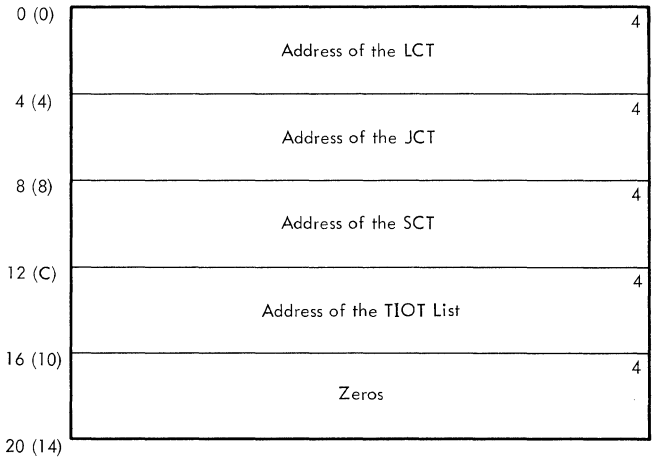


Figure 6. Allocate/Terminate Parameter List

The step initiation routine checks the current step to determine if it is either the checkpoint/restart data set descriptor record (DSDR) processing step or the restart step. If the step is a DSDR processing step being scheduled for a small partition containing less than 12K bytes, the PIB of the partition containing the step initiation routine will be tagged to indicate that the DSDR step is to execute in that partition. The step initiation routine will place the address of its TCB and PIB in the LOT and pass control to allocation via an XCTL macro instruction. If the DSDR step is to be processed in a large partition, normal processing is continued.

If the step is the restart step, the step initiation routine will pass control to partition recovery routine IEFSD518 via a LINK macro instruction. If the return code from IEFSD518 is a zero, normal processing is continued; if the return code from IEFSD518 is a four, the address of the LOT is placed in register 1 and control is passed to job selection IEFSD510 via an XCTL macro instruction.

If the job is using DSO, a message to that effect is placed in the first SMB. Step initiation then passes control to Allocation via an XCTL macro instruction. Allocation returns to IEFSD512 at entry point IEFALRET via an XCTL macro instruction and returns the addresses of a task input/output table (TIOT) list (which points to the TIOT) in the first word of zeros in the APL. On return from Allocation, the return code is tested to determine if allocation was successful. If not, step initiation moves the TIOT to subpool 253 and passes control to alternate step deletion routine IEFSD516 via an XCTL macro instruction.

If allocation was successful, the TIOT is moved to subpool zeor, the ARSA is freed, and the "step started" bit in the SCT is turned on. The address of the job's CSCB is stored in the APL (in the last word of the list). If the job is using DSO, and if job separator and/or system message processing is required, step initiation links to system message and job separator writer routine IEFDSOWR. If IEFDSOWR is unable to process due to a job queue I/O error, the initiator will ABEND with an error code of OBO; if IEFDSOWR is unable to process due to I/O errors, step initiation will set the job failed bit.

The step initiation routine then frees the ARSA and updates the LCT with the TTR of the JCT and the TTR of the SCT. It scans the program properties table (IEFSDPPT) to determine if the program to be executed is to be non-cancellable during its execution. If so, it turns off the cancellable bit in the CSCB. IEFSD512 then uses table breakup routine IEFSD514 to write the TIOT and the LCT on the job queue. Upon return from IEFSD514, the step initiation routine updates the JCT with the TTR of the TIOT. It then uses the queue manager to write the JCT and the SCT back on the job queue, and to read the user parameters into main storage. Finally, it tests to determine if the step is a DSDR processing step and if not, IEFSD512 frees the ECB/IOBs used by the queue manager. It then passes control to problem program interface routine IEFSD513.

Note: For a description of the program properties table, see the MVT Job Management PLM.

SMF User Initiation Exit Routine (IEFSMFIE)

SMF user initiation exit routine IEFSMFIE receives control from step initiation routine IEFSD512. It first determines if SMF is supported by testing the JCTJMROP field of the JCT for a zero value. A zero value indicates that SMF is not supported. In this case the routine immediately

returns control to the caller (IEFSD512). If SMF is supported, the SMF user initiation exit routine performs the following functions:

- It initializes and updates the timing control table (TCT).
- It updates the job log portion of the job management record (JMR).
- It passes control to the user's job initiation exit routine, IEFUJI, or step initiation exit routine, IEFUSI.
- It constructs the SMF Job Commencement Record (type 20).

When it is entered, IEFSMFIE issues the TIME BIN macro instruction and stores the job initiation start time and date in the JCT. It then determines if the step being initated is the first step of the job. If so, it issues a GETMAIN macro instruction specifying the system queue area to obtain main storage for the TCT and for the first 40 bytes of the JMR. The routine initializes the TCT and stores its address in the TCBTCT field of the TCB. It then uses the Queue Management Read/Write routine to bring the JMR into main storage. It copies the first 40 bytes of the JMR into the area reserved for it and updates it with the job initiation start time and date. If user exits are specified, the routine brings the job account control table (ACT) into main storage and then passes control to user job initiation exit routine IEFUJI.

If the step being initated is not the first step of the job, the TCT and JMR are already in main storage. IEFSMFIE stores the step initiation start time and date in the JMR. If user exits are specified, the routine brings the job ACT into main storage and then passes control to user step initiation routine IEFUSI.

Upon return from the user exit routine, IEFSMFIE inspects the return code. If the return code specifies that the job is to be canceled, the routine sets the job-failed bit in the JCT.

For each job, the SMF user initiation exit routine also determines if the data set accounting option is specified by testing the SMCAOPT field in the SMCA. If the option is not specified, or if the job was cancelled, the routine bypasses construction of a Job Commencement Record (type 20). Otherwise, IEFSMFIE builds the record using the accounting information in the job ACT and issues an SVC 83 to have the record transferred to the SMF buffer.

When processing is complete, IEFSMFIE returns control to the caller (IEFSD512).

Note: For the format and description of the JMR and TCT, see "Appendix A" in the MVT Job Management PLM.

Problem Program Interface Routine (IEFSD513)

The problem program interface routine prepares the partition for execution of the job step. It first passes control to SMF TCTIOT construction routine IEFSMFAT. Upon return the routine determines if SMF is supported by testing register 15. A zero value indicates that SMF is not supported and in this case IEFSD513 bypasses the procedures for updating the TCT with the job wait time limit.

If SMF is supported, register 15 contains the address of the TCT and register 0 contains the job wait time limit obtained from the system management control area (SMCA) by IEFSMFAT. In this case IEFSD513 places the job wait time limit in the TCTWLMT field of the TCT. It also initializes bit zero of the TCTSW field to correspond with the bit set in the time remaining field of the timer work area by IEFSD512 indicating whether the job time remaining or the step time limit was established as the time limit for the step about to receive control.

The problem program interface routine then tests to determine if scheduling was performed for a small partition. If so, this routine tests its partition's PIB to determine whether a checkpoint/restart data set descriptor record (DSDR) is to be processed. If the DSDR step is to be processed, the SPIL pointer in the LOT is ignored; otherwise the address of the APL is placed in the SPIL, ECBA in the SPIL is posted to indicate that scheduling is complete, and a WAIT is issued on ECBC. This WAIT allows the small partition module to copy tables and work areas into the small partition. When the tables have been copied, ECBC is posted complete, and the interface routine frees all storage obtained for tables and work areas except for the LOT block, which is retained. The address of the LOT block is placed in register 1 and this routine passes control to job selection, IEFSD510, via an XCTL macro instruction.

If scheduling was not performed for a small partition, a test is made to determine if the job has been canceled. If so, exit is made by issuing an ABEND macro instruction.

If the job has not been canceled, the job OMPA and the SYSOUT QMPA are moved from the LOT to the CSCB, the TIOT is moved to the lowest possible location (subpool 0) in the partition, and a GETMAIN macro

instruction specifying subpool 253 is issued for the user's parameter list (UPL). The UPL (Figure 7) is initialized from the SCT. Another GETMAIN macro instruction (subpool 253) is issued to create a register save area for the user's problem program. If STEPLIB, JOBLIB, and/or FETCH have been specified, their DCBs are created (but not opened) in subpool 253. The JCT, SCT, and APL are now freed, the STEPLIB or JOBLIB and FETCH DCBs are opened, and the TIOT is then moved to subpool 253. A single DCB is used for STEPLIB or JOBLIB, with STEPLIB overriding JOBLIB if both are present.

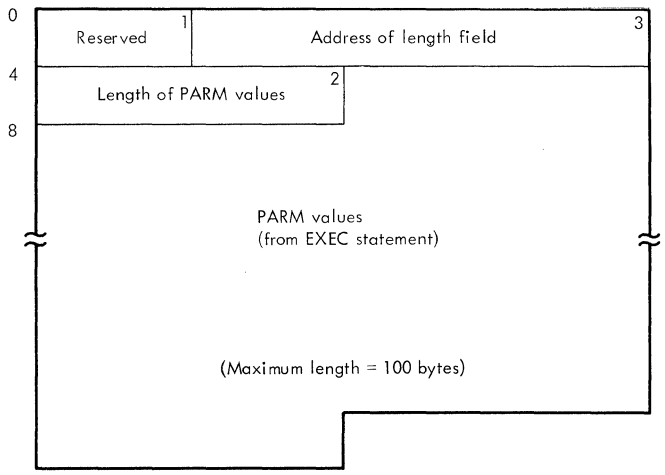


Figure 7. User's Parameter List

If the job being started in the partition is a checkpoint/restart data set descriptor record (DSDR) processing job, the routine bypasses opening the STEPLIB, JOBLIB, and FETCH DCBs and also bypasses setting the storage protection key.

Note: The use of subpools, and the order in which control blocks and tables are created, moved, or deleted, follows a particular sequence even though this handling occurs within different modules. This is done to prevent fragmenting main storage within the partition.

The routine then sets the PSW to the problem program mode. IEFSD513 then tests to determine if job step timing will be performed. If the step time limit in the timer work area of the LOT block is equal to 24 hours, the job step will not be timed. If the step time limit is equal to any value other than a 24 hours, the problem program interface routine issues the STIMER macro instruction to set up the step time interval. It then sets bit zero of the job step timing status bits field in the PIB to one to indicate that the job step TQE is being used by the Initiator. It also sets bit one to one to indicate to step deletion routine IEFSDS15 that the STIMER macro instruction was issued specifying the TQE addressed in the PIB.

Whether or not job step timing is performed, IEFSD513 frees main storage for the LOT block, moves the TIOT to the highest available position within the partition, updates the TCB, and passes control to the problem program via an XCTL macro instruction.

SMF TCTIOT Construction Routine (IEFSMFAT)

If SMF is in the system and if the user accounting option is specified, the SMF TCTIOT construction routine IEFSMFAT builds and initializes a timing control task input/output table (TCTIOT). The routine first determines if SMF is supported by testing the TCBTCT field of the TCB for a zero value. A zero value indicates that SMF is not supported. In this case the routine places a return code of zero in register 15 and returns control to the caller (IEFSD513).

If SMF is in the system, the TCBTCT field contains the address of a TCT built by SMF user initiation exit routine IEFSMFIE. In this case the routine obtains the job wait time limit from the system management control area (SMCA) for return in register 0 to IEFSD513 for updating the TCT. If user exits are specified, IEFSMFAT places the address of SMF user time limit expiration routine IEFUTL in the TCT.

The routine next determines if the user step option is specified by testing the SMCA options field for a X'40'. For any other value the option is not specified and the TCTIOT construction is bypassed. If the user step option is specified, IEFSMFAT constructs a TCTIOT to contain the information necessary for the SMF termination record. The routine issues a GETMAIN macro instruction specifying the system queue area to obtain storage for the TCTIOT and initializes the TCT EXCP counter lookup table.

Whether or not the routine constructed a TCTIOT, it initializes the TCT core map for both hierarchies (0 and 1). It utilizes the boundary box describing the partition to determine the lowest addresses allocated at the high end of hierarchies 0 and 1, and the highest addresses allocated at the low end of hierarchies 0 and 1. It then calculates the amount of storage unused and stores these figures in the TCT.

IEFSMFAT issues a TIME macro instruction to obtain a time stamp to indicate the time the problem program started loading. The time stamp is stored in the TCT.

Finally, IEFSMFAT places the TCT address in register 15 and returns control to the caller (IEFSD513).

Note: For the format and description of the SMCA and the TCTIOT, see "Appendix A" in the MVT Job Management PLM.

Step Deletion Routine (IEFSD515)

Step deletion routine IEFSD515 is entered at the end of step execution to prepare the partition for continued execution of the job, to interface with the termination subroutine, to prepare for the initiation of the next step, or to branch to job deletion if there are no more steps in the current job.

When step deletion is entered, a check is made to determine whether the routine was entered due to an ABEND with the scheduler in control. If so, a message stating that the scheduler has ABENDed is issued to the operator and all CSCBs are removed from the CSCB chain. DSO processing, if any, in the partition is marked for stopping. Control passes to job selection routine IEFSD510 which passes control to DSO stop and modify command processing routine IEFDSOSM.

If the scheduler ABENDs again while trying to stop DSO, the DSOCB will be marked as being no longer available for selection. The DSOCB I/O device will remain allocated to DSO, and the device will not be available until the system is reinitialized.

If an ABEND did not occur, the step deletion routine prepares to calculate the amount of time used by the step and the job when the last step completed execution. It determines if job step timing is being performed by testing the high-order bit in the job step timing status bits field of the PIB. If the bit is off, the following processing is bypassed. If it is on, the TQE is being used for job step timing and IEFSD515 issues a TTIMER macro instruction to stop the timing started by problem program interface routine IEFSD513. It also obtains the step time remaining for use in updating the timer work area when the LOT block is read back in. It then turns off the high-order bit in the job step timing status bits field of the PIB.

Whether or not job step timing is being performed, the step deletion routine branches to ENQ/DEQ purge routine IEFSD598 via a BALR instruction to remove any control blocks which were enqueued, but not dequeued, by the problem program step.

Step deletion then issues a series of GETMAIN requests to obtain storage for queue manager IOBs (IOB1 and IOB2), a temporary QMPA, and a register save area and parameter list for the table breakup routine. These blocks and tables are initialized and step deletion branches to queue manager table breakup routine IEFSD514, to read in the TIOT and LOT blocks for the job step.

IEFSD515 updates the step time remaining field of the timer work area in the LOT block with the step time remaining value obtained from the TTIMER macro instruction. It restores the addresses in the TIOT and LOT blocks, and frees the temporary work areas.

It returns the job QMPA and the SYSOUT QMPA to the LOT block from the CSCB to reflect any activity that occurred during problem program execution.

A GETMAIN (subpool 253) is issued to obtain storage for the SCT and JCT. The SCT is read into storage from the job queue, the JCT from its temporary area. The JCT is updated with the address of the next SCT and written back on the job queue.

A test is made to determine if job step timing is being performed. If the step time limit in the timer work area is equal to 24 hours and if bit one of the job step timing status bits field in the PIB is set to zero, neither the job nor the step has been timed and the routine bypasses the following processing and obtains storage for the terminate register save area and parameter list. If the step has been timed, the values in the timer work area must be updated to reflect the time used by the step that just completed execution. If SMF is supported (determined by a nonzero value in the TCBTCT field of the TCB), the time extension specified is calculated and added to the step time limit.

The information in the timer work area is then used to calculate the new values for job time used, job time remaining, step time used, and step time remaining, and the timer work area is updated with these calculations.

The Queue Management Read/Write routine is used to read in the job and step ACTs. They are updated with the new values for the job and step time, and then written back out.

Storage is obtained for a terminate register save area and a terminate parameter list. The terminate parameter list is initialized with addresses of control blocks (LOT, JCT, SCT, and TIOT list) and the step deletion routine branches to the termination subroutine via a BALR instruction. When termination

returns control, step deletion frees the terminate register save area and terminate parameter list and then reinitializes the WTPCB for the next step of the job or the next job. If the partition was executing the DSDR step for a small partition, step deletion places the addresses of the small partition's TCB and PIB into the LOT block. Step deletion then checks the return code from termination.

If the return code indicates that the job is to be suspended, step deletion loads the address of the LOT block in register 1. In MFT systems with the 44K scheduler, step deletion then passes control to IEFSD168 via a BALR instruction. In MFT systems with a 30K scheduler, however, step deletion branches to linkage routine IEFSD167 to pass control to IEFSD168 via an XCTL macro instruction. If the return code indicates that job termination was entered, step deletion branches to job deletion routine IEFSD517 and, in MFT systems with the 44K scheduler, receives control again. In MFT systems with the 30K scheduler, however, control does not return to step deletion. It is passed immediately to IEFSD517. If job termination was not entered, the SCT for the next step of the job is read from the job queue, and step deletion passes control to IEFSD512 via an XCTL macro instruction.

Note: If a small partition is requesting termination, entry to the step deletion routine is made at special entry point SMALTERM. When the routine is entered at this point, it performs the following functions before invoking ENQ/DEQ purge routine IEFSD598. It obtains the step time remaining for the step which executed in the small partition from the SPIL and saves this value for updating the step time remaining field of the timer work area in the LOT block, when the block is read back in. IEFSD515 also establishes pointers to the SPIL and the small partition's TCB.

ENQ/DEQ Purge Routine (IEFSD598)

At job termination, this routine purges all ENQ/DEQ control blocks associated with the TCB address passed in Register 4 by the caller. If step termination was completed instead, this routine purges all ENQ/DEQ control blocks except the data set integrity blocks associated with the major name SYSDSN.

When a given resource is dequeued for the subject TCB, a task switch may occur for a higher priority requestor whose wait count becomes zero, due to availability of the resource. (This purge routine operates in a disabled state to prevent concurrent updating of the ENQ/DEQ control blocks.)

Alternate Step Deletion Routine (IEFSD516)

Alternate step deletion routine IEFSD516 is entered from step initiation routine IEFSD512 when allocation for a step has not been successful. Using the APL and ARSA (created by the step initiation routine) as the terminate parameter list and terminate register save area, this routine branches to termination subroutine IEFSD22Q via a BALR macro instruction. When control is returned from termination, the storage used for the parameter list and register save area is freed and a test is made to determine if job termination was entered. If so, this routine branches to job deletion routine IEFSD517. If job termination was not entered, the SCT for the next job step is read from the job queue and this routine branches to step initiation routine IEFSD512.

Job Deletion Routine (IEFSD517)

The job deletion routine is called at job termination to delete the job from the input queue and to prepare the partition for initiation of the next job. The routine sets the high-order byte of the LCTTCBAD field of the LCT to '80' (hexadecimal) to indicate to the ENQ/DEQ purge routine that it is job termination instead of step termination. The routine then branches to ENQ/DEQ purge routine IEFSD598 to purge the control blocks. On return from the purge routine, the high-order byte is reset to '00'.

The job deletion routine then deletes the job from the input queue, using queue manager delete routine IEFQDELQ. All areas of storage in the partition which were used for the job (except the LOT block) are freed, and the job's CSCB is freed by issuing an SVC 34. The PIB fields used for the disk address of the TIOT and the LOT block are set to zero. If termination was for a small partition, ECBA in the SPIL is posted with a code of two (indicating job termination for the small partition). If termination was for a large partition (or after ECBA has been posted) the "no work" ECB in the PIB is posted and the job deletion routine branches to job selection routine IEFSD510.

Partition Recovery Routine (IEFSD518)

Partition recovery routine IEFSD518 determines the location of main storage required for a checkpoint restart. If the partition being scheduled for the job to be restarted contains the required main storage, the JCT is checked to determine if the job used DSO. If it did, the job's SIOTs are checked to determine which types of I/O devices were used. If any needed type is not available, a message informing

the operator of the missing devices is sent and the job is placed on the hold queue. If all devices are available, the routine returns to the step initiation routine for normal processing. If the nucleus has expanded past the lower boundary of the partition containing the required main storage, the routine sets the job fail bit in the JCT, issues a message stating that main storage is not available for the job, and returns to the step initiation routine IEFSD512 with a return code of zero.

If the partition being scheduled does not contain the required main storage, the routine places the job on the hold queue, updates the SCD and places the SCD back on the job queue. The job's CSCB is unchained and the space containing the CSCB and the ECB/IOBs is freed. If the job used DSO, the routine links to release DSOCB routine IEFDSOFB to release any DSO processor allocated to the job. The routine then branches to ENQ/DEQ purge routine IEFSD598.

Upon return from ENQ/DEQ purge routine, if a problem program partition exists that contains the required main storage, this routine will create an internal queue element and chain it to the partition's PIB. The partition's "no work" ECB will be posted and a message will be issued stating that the job will start in the partition. If an existing partition contains the required main storage and is defined as a reader or writer partition, this routine issues a message indicating that the partition must be redefined to accept the desired jobclass. If no partition contains the required main storage or the partition that contains the required main storage is about to be redefined, this routine issues a message stating the length and displacement of the required main storage. If the partition being scheduled was a large partition its no-work ECB is posted; if it was a small partition, the SPIL is posted indicating job termination. The partition recovery routine frees the JCT and SCT areas of the partition and returns control to step initiation routine IEFSD512 with a return code of four.

Dequeue by Jobname Interface Routine (IEFSD519)

Dequeue by jobname interface routine (IEFSD519) builds a parameter list used by dequeue by jobname routine IEFLOCDQ to locate a job named on the checkpoint/restart internal job queue. When a checkpoint/restart job is indicated by an entry in the internal job queue pointer in the PIB being processed by job selection routine IEFSD510, job selection branches to IEFSD519 which builds the seven-word parameter list required by IEFLOCDQ. When the job is dequeued, IEFLOCDQ returns control to IEFSD519.

The interface routine marks the job as ready and returns to job selection with a code of zero in register 15, indicating that the job has been found, and a pointer to the LOT in register 1. If the job is not found by IEFLOCDQ, a return code of four is returned in register 15 to job selection. (For a description of IEFLOCDQ see the MVT Job Management PLM.)

System Output Writers

MFT uses the MVT system output writer (Charts 15-16) with minor changes to five of the modules. As in MVT, the user may have up to 36 system output writers operating concurrently in the system. Each output writer can handle eight output classes; output classes may be shared by writers. However, in MFT, system output writers are classified as either resident or nonresident. A resident writer operates in its own partition. A nonresident writer operates in any problem program partition large enough to accommodate it.

RESIDENT WRITERS

Resident output writer partitions are designated in the TCB by a setting of '10' in the first two bits of the pointer to the partition information block (PIB). This designation is made at system generation by assigning W to the partition in place of the job class or by redefining a partition and assigning WTR to it.

A resident writer is activated by issuing a START command specifying a partition designated previously as a writer partition. A resident writer can be terminated only by issuing a STOP command specifying the device assigned to that writer.

NONRESIDENT WRITERS

A nonresident system output writer may be started in a problem program partition large enough to hold the writer by issuing a START command specifying either that partition or by replacing the partition number with an 'S' to specify a system-assigned nonresident writer.

When the writer has started, it executes in the same way as a resident writer and must be terminated by a STOP command to allow processing of problem programs to be resumed in the partition.

SYSTEM OUTPUT WRITER MODULES

The following five MVT system output writer modules are modified for MFT.

- IEFSD070 - Data Set Writer Linkage Routine.
- IEFSD079 - Linkage to Queue Manager Delete Routine.
- IEFSD084 - Wait Routine.
- IEFSD085 - Data Set Block (DSB) Handler Routine.
- IEFSD087 - Standard Writer Routine.

Descriptions of all other system output writer modules can be found in the MVT Job Management PLM.

Data Set Writer Linkage Routine (IEFSD070)

This routine passes control to the appropriate writer routine via a LINK macro instruction. The normal linkage is to the standard writer, IEFSD087. If a special user-written output writer routine is requested, this routine passes control to that writer. Upon return from either writer, the routine passes control to data set delete routine IEFSD171 via an XCTL macro instruction which deletes the output data sets from the output queue.

Linkage to Queue Manager Delete Routine (IEFSD079)

Upon completion of a job, linkage module IEFSD079 passes control to queue manager delete routine IEFQDELQ via an XCTL macro instruction to delete all control blocks and SMBs associated with the output job from the job queue. Following deletion, the routine then posts all reader ECBs that are waiting for space to indicate that space is now available. (The reader ECB chain address is obtained from the master scheduler resident data area.) When all ECBs have been posted, control is returned to main logic routine IEFSD082.

Wait Routine (IEFSD084)

This routine serves as a multiple WAIT when there is no work in any of the output classes associated with the writer. It issues a WAIT macro instruction on the ECB list created by class name setup routine IEFSD081. When the system output writer enters a wait state, the wait routine issues a message informing the operator that the writer is waiting for work. Any posting (such as a command, or work for the writer) causes control to be given to IEFSD082.

DSB Handler Routine (IEFSD085)

DSB handler routine IEFSD085 is the setup module for printing data sets. It issues a GETMAIN macro instruction for the input DCB if it was not obtained before, and constructs a new TIOT containing an entry for the input data set. It also sets up any user-written output writer program. A check is then made to determine if a pause is required between data sets or only at forms change. If a special form is to be used, the routine writes a message to the operator telling him what form to put in the output device. The form change only occurs if the output device is unit record. This routine then passes control to linkage routine IEFSD070 via an XCTL macro instruction.

Standard Writer Routine (IEFSD087)

This routine first issues an OPEN macro instruction to open the output data set. If the data set was not opened by the problem program, no attempt is made to process the data set. After OPEN, a test is made to check for machine control characters. A switch is set that is interrogated by PUT routine IEFSD089. The writer then passes control to transition routine IEFSD088 which creates header and trailer records. Upon return from IEFSD088, the writer routine checks the CANCEL ECB in the CSCB to determine if a CANCEL command has been issued for this writer. If the CANCEL ECB has been posted complete, control passes to transition routine IEFSD088 to create a trailer record. When control is returned from IEFSD088, the writer is closed. Control is then returned to linkage routine IEFSD078 via a RETURN macro instruction.

If the writer is not to be canceled, the writer routine issues a GET macro instruction to read a record and checks for a control character. If no control character exists, the writer puts one in which causes the printer to skip one line or the punch to feed into the normal pocket. If the printer has overflowed, a skip is made to the next page.

The writer then adjusts the pointer to the record so that it points to the first data character (instead of control character) and passes control to transition routine IEFSD088 for trailer records. It then issues a CLOSE macro instruction to close the input data set, a FREEPOOL macro instruction to free the buffers, and returns control to linkage module IEFSD078 via a RETURN macro instruction.

Direct System Output Processing

Direct system output (DSO) processing operates in MFT in the same manner as in MVT. The main difference between DSO in MFT and MVT is that DSO started in an MFT partition can only process output from jobs within that partition whereas DSO started in an MVT system is not restricted by partition boundary.

PART 3: COMMAND PROCESSING

Operator commands control system operation and modify system tasks. Command processing in MFT is handled by the SVC 34 command scheduler routines, the master scheduler resident command processor routines, and the system task control routines. With the exception of DEFINE, HALT, MODE, and SWAP, commands can be entered into the system through the console or the input job stream. The DEFINE, HALT, MODE, and SWAP commands can be entered only through the console. Commands entered through the console are read by the communications task and routed to the master scheduler. When a command is encountered in the input stream, the reader/interpreter passes control to SVC 34 to process the command. SVC 34 processes most commands completely and returns control to the interrupted routine.

The commands accepted and processed by MFT are the following:

CANCEL
CONTROL
DEFINE
DISPLAY
DUMP
HALT
HOLD
LOG
MODE
MODIFY
MONITOR
MOUNT
MSGRT
RELEASE
REPLY
RESET
SET
START
STOP
STOPMN
SWAP
SWITCH
UNLOAD
VARY
WRITELOG

The format and syntax of these commands can be found in the Operator's Reference manual.

SVC 34 processes all commands completely except CANCEL, DEFINE, DISPLAY (A, CONSOLES, jobname, N, Q, U), HOLD, LOG, RELEASE, RESET, START and WRITELOG. SVC 34 does preliminary processing of these commands and passes control to the master scheduler resident command processor to complete the processing of all but the START, WRITELOG, and LOG commands. If the master scheduler resident command processor is processing a DEFINE command, SVC 34 queues all commands until the DEFINE command has been completely processed.

When a WRITELOG command is found, SVC 34 stores it and posts the System Log task ECB. (See the MFT Supervisor PLM.) When a START command is found, SVC 34 builds and chains a CSCB, places the address of the CSCB in the partition's PIB, and posts the partition. The system task control routines further process the START command. When a LOG command is found, SVC 34 issues a WTL macro instruction (SVC 36) to have the LOG command processed in a manner similar to a write-to-log macro instruction issued by a problem program.

When processing commands, interruptions are disabled so that command processing may be completed before any other interruptions are serviced. Although commands are processed when issued, the command may not take effect immediately. An example of this is the STOP writer command. The master scheduler marks a command scheduling control block (CSCB) which is checked by the writer between jobs. The command does not take effect until the writer completes the job it was processing when the command was issued.

SVC 34 Routines

SVC 34 (Chart 6) is called to process all commands. As previously noted, it processes some of these commands completely and calls the resident command processor or system task control to process the remaining commands. The commands processed completely by SVC 34 are:

CANCEL (active jobs only)
CONTROL
DISPLAY (R, SQA, T)
HALT
MODE
MODIFY
MONITOR (DSNAME, JOBNAMES, SPACE, STATUS)
MOUNT
REPLY
STOP
STOPMN
SWAP
SWITCH
UNLOAD
VARY

The SWAP command is accepted and processed only if Dynamic Device Reconfiguration (DDR) is in the system.

There are four SVC 34 routines that are unique to the MFT configuration of the control program. These routines include the DEFINE and MOUNT Commands routine (IEESD571), the CANCEL command routine (IEE2803D), and the STOP INIT and START commands processing routines (IEESD561 and IEE3903D). These routines are described below.

Other SVC 34 routines are described in the MVT Job Management PLM. Two major differences between SVC 34 processing in MFT and MVT should be noted:

- TSO commands are processed by the MVT SVC 34 routines. TSO is not supported by the MFT configuration of the control program.
- A STAE environment is established for the MVT SVC 34 routines, but not for the MFT SVC 34 routines. Therefore, in MVT, the first SVC 34 routine to be executed creates the STAE environment. This routine precedes chain manipulation routine IEE0303D, which is the same in both systems, and is the first SVC 34 routine to be executed in MFT.

DEFINE and MOUNT Routine (IEESD571)

This routine processes the DEFINE command by setting the necessary indicators in the master scheduler resident data area. It then posts the ECB for the master scheduler resident command processor IEECIR50.

This routine processes the MOUNT command as that command is processed in PCP. It builds a parameter list for, and issues an XCTL macro instruction to, the PCP master command EXCP routine IGC0103D.

CANCEL Command Routine (IEE2803D)

This routine processes the CANCEL command by scanning the CSCBs for the job name given in the CANCEL command. If the job name is found, indicating that the job is active, and if the command did not have an IN or OUT parameter, the CSCB is checked to determine if it is cancelable, that is, if it represents a problem program. If it does, IEE2803D issues a BALR to ABTERM, passing the address of the job's TCB and indicating a completion code of 222 if no dump is to be taken, or 122 if a dump is to be taken.

If the CSCB is not cancelable, that is, if it represents a system task, the CSCB is marked canceled and is posted.

If the job is represented on the CSCB chain, but the command specified IN or OUT, the "Job Selected" message is written to the operator and control is returned to the caller.

If the job is not represented on the CSCB, indicating that the job is either in the input or output queue(s) or that it does not exist, IEE2803D passes control via an XCTL macro instruction to CSCB creation routine IEE0803D to build a CSCB for the CANCEL command. (See the MVT Job Management PLM for a description of IEE0803D.)

STOP INIT and START Commands Processing Routines (IEESD561 and IEE3903D)

These routines perform the initial processing for all the START commands and the STOP INIT command. When a START command is received, STOP INIT and START command syntax scan routine IEESD561 examines the command parameters. If anything other than a system reader or writer is to be started, the routine determines the number and status of the partition named in the command. If the command is a STOP INIT command, IEESD561 determines which partition contains the initiator to be stopped. The routine then passes control via an XCTL macro instruction to STOP INIT and START Command Processor routine, IEE3903D. (See Figure 8.)

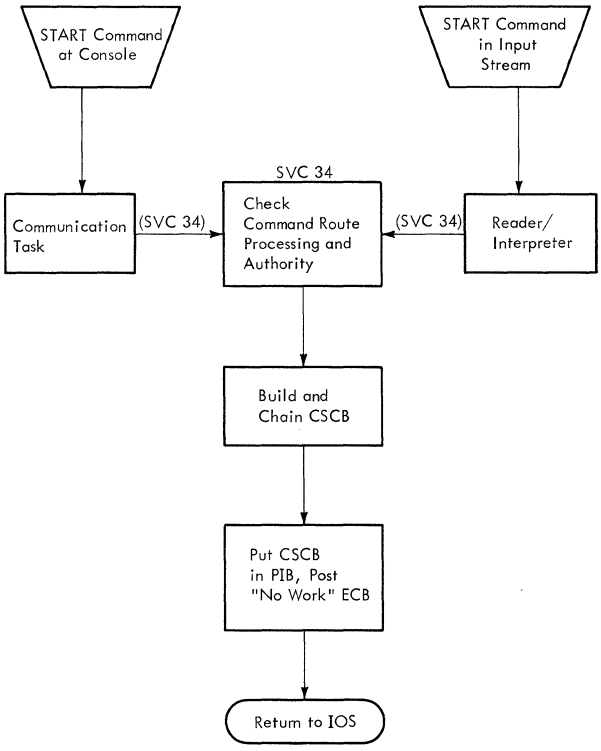


Figure 8. START Command Processing Flow

If the command is a START command, command processor routine IEE3903D builds and chains a CSCB, places the address of the CSCB in the partition's PIB, and posts the partition. If a system reader is to be started, the routine searches for a scheduler-size problem program partition which is inactive; if a system writer is to be started, the routine searches for any inactive problem program partition. If a partition is located, the routine builds and processes a CSCB as stated above. If a partition cannot be found, the routine issues a message to the operator stating that the command has failed. If the command is a STOP INIT command, the routine verifies that the partition contains an initiator and sets the STOP INIT indicator in the partition's PIB.

The section "System Task Control" describes the further processing of the START command CSCB. The processing of a STOP INIT indicator is completed by the Initiator/Terminator.

Master Scheduler Resident Command Processor

In MFT systems, SVC 34 does preliminary processing for the commands CANCEL, DEFINE, DISPLAY (A, CONSOLES, jobname, N, PFK, (C,K),Q U), DUMP, HOLD, MONITOR A, RELEASE and RESET, and passes control to the master scheduler resident command processor, IEECIR50, to complete the processing. IEECIR50 in turn passes control via a LINK macro instruction to the Command Analyzer routine, IEECIR51. IEECIR51 analyzes the command and passes control to the Queue Alter routine, to the DEFINE command processor, or returns control to IEECIR50 for issuing of SVC 110.

The master scheduler resident command processor resides in the nucleus and operates under control of its own TCB. The master scheduler TCB is always dispatchable and is of higher priority on the TCB queue than the TCBs for the partitioned area (the problem program area) of main storage.

The master scheduler resident command processor, IEECIR50, waits on an ECB which is posted by SVC 34 when a command has been scheduled for processing. Control is then passed to the command analyzer routine, IEECIR51, which scans the CSCB chain for any commands to be processed. If a command is found, the CSCB is removed from the chain and control is passed to the appropriate routine:

- For D PFK, D C,K, D U, and DUMP, control is returned to IEECIR50, which will issue SVC 110 to pass control to the Master Scheduler Router for further processing.
- For all other commands having a CSCB, control is passed to the Queue Alter Syntax Check routine (IEESD562) via an XCTL macro.
- For DEFINE, control is passed to IEEDFIN1 via XCTL.

MASTER SCHEDULER ROUTER ROUTINE

In MFT systems, the Master Scheduler Router routine, module IEE00110 (which receives control when SVC 110 is issued), receives control from IEECIR50 to continue processing of DISPLAY U, DISPLAY C,K, DISPLAY PFK and DUMP commands. The Master Scheduler Router routine determines which command has been entered and passes control accordingly:

- If the command is D U, control is passed to the DISPLAY Units routine, IEE20110.
- If the command is D PFK, control is passed to the DISPLAY PFK routine, IEE40110.
- If the command is D C,K, control is passed to the DISPLAY C,K routine, IEE10110.
- If the command is DUMP, control is passed to the DUMP processor, IEE60110.

When processing by SVC 110 is complete, control is returned to the master scheduler resident command processor.

QUEUE ALTER ROUTINE

The master scheduler uses the queue alter routine (Chart 3) to handle the execution of commands that require access to or manipulation of system queues. The commands processed by the queue alter routine include: CANCEL, DISPLAY (A, CONSOLES, jobname, N, Q), HOLD , MONITOR A RELEASE and RESET.

The queue alter routine is a collection of several modules that check command syntax, manipulate system queues and control blocks, and issue informational messages to the operator. It is entered via an XCTL macro instruction issued by the master scheduler command analyzer, IEECIR51, after it has deleted the command CSCB from the CSCB chain.

The queue alter routine is common to both MFT and MVT systems with the following exceptions:

- In MFT systems the queue alter routine receives control from IEECIR51 (a

module used only by MFT), via an XCTL macro instruction, and returns control to IEECIR50. In MVT systems the routine receives control from a module used only by MVT, via an ATTACH macro instruction, and returns control to the MVT module.

- In MFT systems the queue alter routine operates under control of the master scheduler TCB and is therefore part of the master scheduler task. In MVT systems the routine operates under control of its own TCB.

- In MFT systems queue alter syntax check routine, IEESD562, tests for DISPLAY (A and CONSOLES) and MONITOR commands and passes control to the appropriate command processing routines. In MVT systems the processing of these commands is in no way associated with the queue alter routine.

Because of these differences, the queue alter routine modules and the DISPLAY A and DISPLAY CONSOLES processing routines are described below, even though they are common to both systems.

Syntax Check Routine (IEESD562)

Syntax check routine IEESD562 checks the syntax of the command parameter in the CSCB. If a search of the input work queues (SYS1.SYSJOBQE) is required for processing the command, the syntax check routine sets internal codes for the queue search and passes control to the ECB/IOB construction routine, IEESD582. If the command syntax is in error, control is passed to the service routine, IEESD565. If the command was a DISPLAY A or MONITOR A command, control is passed to the DISPLAY A routine, IEESD566. If the command was a DISPLAY CONSOLES command, control is passed to the DISPLAY CONSOLES routine, IEEXEDNA.

DISPLAY A ROUTINE (IEESD566): DISPLAY A routine IEESD566 receives control from syntax check routine IEESD562 when the DISPLAY A (active) command MONITOR A (active) command is entered. This routine constructs WTO messages containing the active job and stepnames and, if subtasking is included, a count of the number of subtasks within the job step. The DISPLAY A routine returns control to the master scheduler resident command processor.

DISPLAY CONSOLES ROUTINE (IEEXEDNA): DISPLAY CONSOLES routine IEEXEDNA receives control from the Syntax Check routine IEESD562 when the DISPLAY CONSOLES command is entered. This routine issues a header message that describes the status message. It then constructs and issues a message describing the status of the hard copy log (if one exists) and each console in the system, both active and inactive. When the message is issued, it returns to IEECIR50.

DISPLAY U ROUTINES (IEE20110, IEE21110, IEE22110 AND IEE23110): The resident command analyzer routine (IEECIR50) issues an SVC 110 to handle processing of DISPLAY U commands. The master scheduler router routine (IEE00110), which is the first routine of SVC 110, passes control to IEE20110. This routine checks the command for syntax errors and if any are found, passes control to IEE22110 to issue an error message. If there are no errors, control passes to IEE21110 which assembles the display and passes control to IEE23110. This routine issues a WTO macro instruction to write the display to the operator console. When all lines have been written, IEE22110 issues a FREEMAIN macro instruction to free work areas and then returns control to the master scheduler router.

DISPLAY PFK ROUTINE (IEE40110): The resident command analyzer routine (IEECIR50) issues an SVC 110 to continue processing of a DISPLAY PFK command. The master scheduler router (IEE00110), which is the first routine of SVC 110, passes control to IEE40110. This routine checks the command syntax, assembles the display of commands associated with each PFK key number, and issues a WTO macro instruction to pass the display to the operator console. When processing is complete, control is returned to the master scheduler router.

DISPLAY C,K ROUTINES (IEE10110, IEE11110, AND IEE12110): The resident command analyzer routine (IEECIR50) issues an SVC 110 to continue processing of a DISPLAY C,K command. The master scheduler router (IEE00110), which is the first routine of SVC 110, determines that a D C,K command was entered and passes control to the DISPLAY C,K routine (IEE10110). This routine begins to assemble the display of CONTROL command operands, and issues a WTO macro instruction to pass the display to the operator. The second DISPLAY C,K routine (IEE11110) and the third routine (IEE12110) continue assembling the display and issuing WTO macro instructions until the entire display is passed to the operator. When processing is complete, control passes back to the master scheduler router.

Queue Search Setup Routine (IEESD563)

The queue search setup routine determines which of the queues is to be searched and reads the queue control record (QCR) for that queue. If the queue must be searched, parameters for the search are established.

- If the DISPLAY Q or DISPLAY N command is being processed and this is the initial entry to IEESD563, control passes to the DQ/DN message setup routine, IEESD584.

- If the DISPLAY Q command is being processed, and an empty held queue is found, no further queue searching is needed, so control passes to the Queue Search Return Routine.

- If all the queues selected are empty, or for a jobname search with no match found, control passes to the Service Routine, IEESD565.

- If the CANCEL command is being processed with ALL or OUT specified and if all the output queues have been selected and at least one match found, control passes to the Queue Scratch Setup Routine, IEESD575.

- If the HOLD Q or RELEASE Q commands are being processed, the HOLD queue bits in the ACR are adjusted, the ACR rewritten into SYS1.SYSJOBQE and control passes to the service routine.

- For all other commands, control passes to the Queue Search Routine, IEESD564, to begin or to continue the search.

Queue Search Routine (IEESD564)

Queue search routine IEESD564 reads the entries of a queue based on the parameter information passed by setup routine IEESD563. If the command processing requires changes in the chaining information in a queue entry or control record, the updated information is written on the queue. If the DISPLAY N command is being processed, the name of each job and the queue in which it is found is written to the operator, and control is passed to the Queue Search Setup Routine to establish parameters for the next search. For all other commands, control is passed to the Queue Search Return Routine, IEESD583, to analyze and establish action indicators.

Service Routine (IEESD565)

Based on the information passed by the calling routine, service routine IEESD565 performs the following:

1. Passes control to queue manager enqueue routine IEFQMNQQ via a LINK macro instruction to enqueue an entry or QCR.

2. Issues a FREEMAIN macro instruction to free the ECB/IOB which was used to read SYS1.SYSJOBQE.

3. Passes control to the SVC 34 message module (IEE0503D) via a LINK macro instruction to write a message.

4. If another queue needs to be searched, it passes control to queue search set up routine IEESD563 via an XCTL macro instruction.

After the requested processing has been performed, the service routine transfers control to router routine IEECIR50.

Queue Scratch Setup Routine (IEESD575)

Queue scratch setup routine IEESD575 builds the parameter list for the SCRATCH macro instruction (SVC 29) according to whether the canceled job was found on the input or output queue(s). If the job was found on the input queue, IEESD575 determines whether there are SYSIN data sets to be scratched. If not, IEESD575 passes controlto queue alter delete routine IEESD576. If the job was found on the input or output queue with data sets to be scratched, IEESD575 passes control to queue scratch routine IEESD581. When IEESD581 has scratched all data sets, IEESD575 passes control to queue alter delete routine IEESD576.

Queue Alter Delete Routine (IEESD576)

Queue alter delete routine IEESD576 passes control to queue manager delete routine IEFQDELE to delete the queue entries associated with the canceled job. Upon return from the delete routine, control is passed to the queue message class setup routine, IEESD578.

Queue Restart Enqueue Routine (IEESD577)

Queue restart enqueue routine IEESD577 passes control to the queue manager enqueue routine IEFQMNQQ to enqueue the SYSOUT data sets for canceled restarting jobs. Upon return from IEFQMNQQ, IEFSD577 passes control to IEESD579.

Queue Message Class Setup Routine (IEESD578)

Queue message class setup routine IEESD578 zeroes out the DSBs in the message class and sets up the queue manager parameter area for enqueuing the message class. If the job was a restarting job, IEESD578 passes control to IEESD577. For a job on the output queue, with more queues to be searched, control is passed to IEESD563. If the CANCEL command was issued for a job that does not have the message class associated with it control is passed to message routine IEESD580 otherwise control is passed to queue SMB routine IEESD579.

Queue SMB Routine (IEESD579)

Queue SMB routine IEESD579 places the appropriate cancel message into the first SMB and passes control to the queue manager enqueue routine IEFQMNQQ to enqueue the message class. IEESD579 issues the cancel message to the operator and returns control to master scheduler resident command processor IEECIR50.

Message Routine (IEESD580)

Message routine IEESD580 issues cancel messages to the operator for those jobs that do not have the message class associated with them. It then returns control to master scheduler resident command processor, IEECIR50.

Queue Scratch Routine (IEESD581)

Queue scratch routine IEESD581 issues the SCRATCH macro instruction (SVC 29). Upon return from the Scratch service routine, IEESD581 issues a "data set not deleted" message if the return code is nonzero. IEESD581 returns control to queue scratch setup routine IEESD575.

ECB/IOB Construction Routine (IEESD582)

If the operand of the command contains a jobname, the ECB/IOB construction routine searches the chain of CSCB's for the corresponding CSCB. If a match is found, control passes to the service routine for the "job selected" message to be issued. If no match is found or if the operand of the command does not contain a jobname, a GETMAIN is issued to obtain storage and to construct an event control block, (ECB), and an input/output block, (IOB). Control then passes to the queue search setup routine, IEESD563.

Queue Search Return Routine (IEESD583)

The queue search return routine obtains control from the Queue Search Routine, IEESD564, after a queue search has been completed.

- If the DISPLAY jobname command is being processed and a jobname match occurs, the job status information is written to the operator.
- If the DISPLAY Q command is being processed, the queue status and number of entries found in the search by IEESD564 is written to the operator.
- If the CANCEL command is being processed and a jobname match occurs, control is passed to the queue scratch setup routine, IEESD575. The queue search return routine determines whether further queue searching is to be performed. If so, control passes to the Queue Search Setup Routine, IEESD563, to establish the new search parameters. If the queues to be searched are exhausted, control passes to the service routine to setup the appropriate message to the operator.

DQ/DN Message Setup Routine (IEESD584)

The DQ/DN message setup routine writes the control and label lines for the queue or name display in response to a DISPLAY Q or DISPLAY N command. It then passes control to the queue search routine, IEESD564, to begin the search, unless the command was a DISPLAY Q and a empty field queue was found, in which case no search is needed, so control is passed to the search return routine, IEESD583.

DEFINE COMMAND PROCESSOR

The master scheduler resident command processor uses the DEFINE command processing routines (shown in Figure 9) to initialize or change partition definitions in MFT. These routines handle:

- Commands from the operator via a console, issued after nucleus initialization, to change the size and description of any partition while processing continues in unaffected partitions.
- Commands from the system at IPL time to prepare the partition as it was described at system generation.

All transfers of control among the processing routines are accomplished via an XCTL macro instruction.

DEFINE Command Initialization Routine (IEEDFIN1)

The master scheduler passes control to DEFINE command initialization routine IEEDFIN1 whenever a DEFINE command is entered by the operator. The routine also receives control from the master scheduler during system initialization, after the nucleus initialization program (NIP) completes its preparation of the system. In either case the routine builds the DEFINE data area containing the size and description (job classes A-O, or R or W) of each partition If Main Storage Hierarchy Support is included in the system, the data area contains the size of the partitions in terms of hierarchies. Hierarchy 0 represents processor storage and hierarchy 1 represents 2361 Core Storage.

If the time-slicing feature is included in the system, the data area also contains a doubleword of time-slicing information, including the first and last partition numbers in the time-slicing group and the time interval (in milliseconds) assigned to the group of partitions. This data is used at completion of DEFINE processing to define the partitioning of main storage.

If the DEFINE command initialization routine was entered as the result of a DEFINE command, the routine issues a DEFINE COMMAND BEING PROCESSED message to all active consoles. It then determines whether LIST was specified and if so, passes control to listing routine IEEDFIN4. If not, the routine passes control to message routine IEEDFIN5 for issuance of an ENTER DEFINITION message.

If the DEFINE command initialization routine was entered during the system initialization, the routine also issues a DEFINE COMMAND BEING PROCESSED message to all active consoles. It then determines whether partition redefinition or LIST was specified by the operator, and if not, passes control to validity check routine IEEDFIN3. If either LIST or partition redefinition was specified, the routine continues processing as if a DEFINE command had been entered by the operator.

Syntax Check Routine (IEEDFIN2)

When syntax check routine IEEDFIN2 receives control at primary entry point IEEDFIN2, it translates the statements entered by the operator to upper case. When the routine receives control at secondary entry point IEEDPART, this operation is bypassed.

The statement is scanned and each entry in the statement -- a partition definition, a time-slicing change, or a keyword -- is processed separately.

If the entry is a partition definition, the routine checks the entry for syntax errors. If a syntax error is found, the routine passes control to message routine IEEDFIN5 for issuance of the appropriate syntax error message. The erroneous entry and all following entries are ignored. If the syntax is correct, IEEDFIN2 updates the DEFINE data area with the partition information and gets the next entry for processing.

If the entry is a time-slicing change, the routine passes control to time-slice check routine IEEDFIN6.

If the entry is neither a partition definition, nor a time-slicing change, the routine assumes that it is a keyword and passes control to keyword scan routine IEEDFIN7.

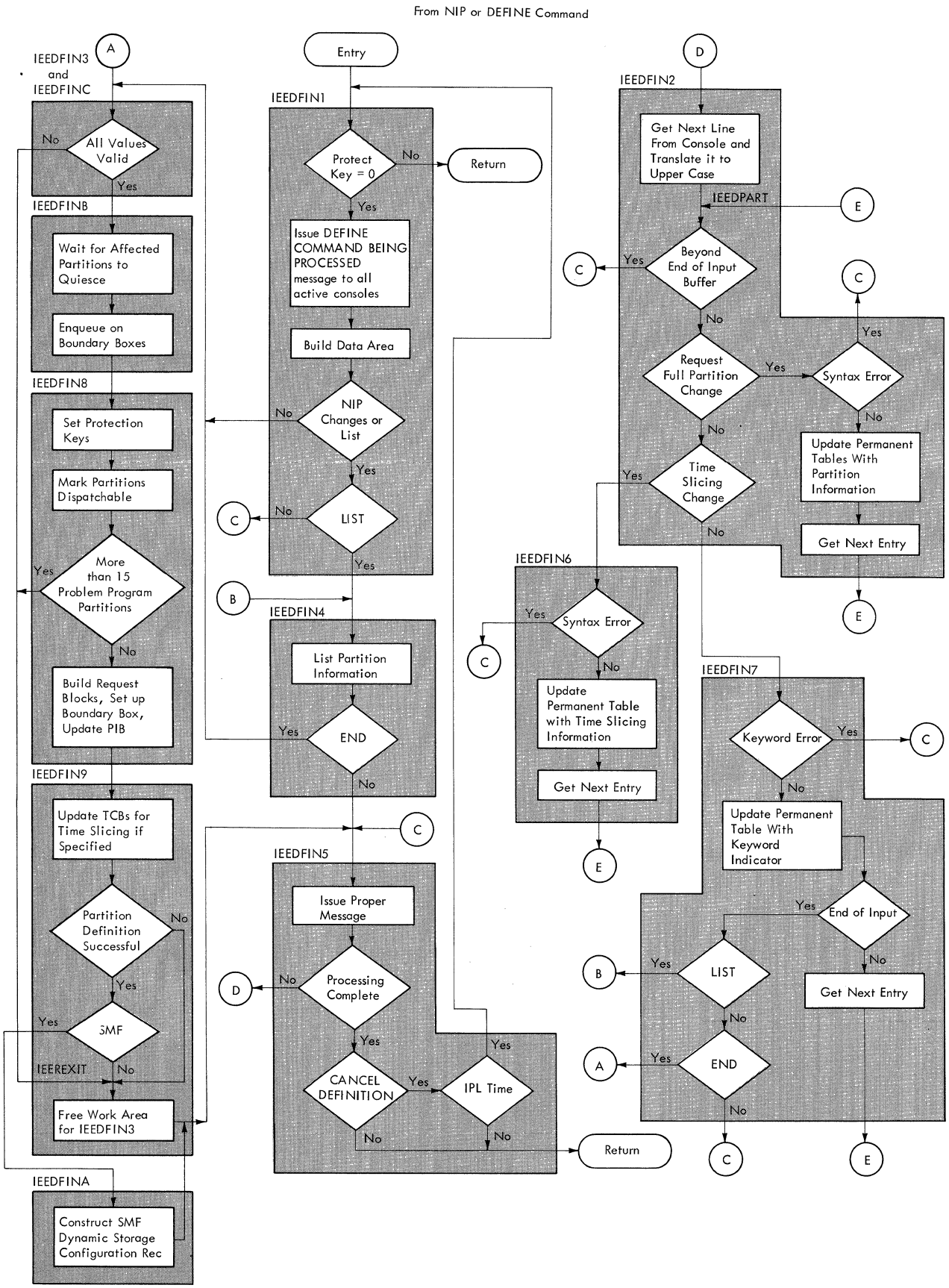


Figure 9. DEFINE Command Processing Flow

Validity Check Routine -- Processor Storage (IEEDFIN3)

Validity check routine IEEDFIN3 (for processor storage) makes final checks to determine whether the information entered by the operator is correct (e.g., that the definition changes which have been requested are within legal bounds or that the time-slicing specification is valid). If an error is detected, the routine passes control to IEEREXIT, a secondary entry point in command final processor routine IEEDFIN9. If the information is valid, the routine determines the partitions affected by the DEFINE command constructs a list of PIB pointers (one for each affected active partition) and passes control to validity check routine IEEDFINC.

Validity Check Routine -- Core Storage (IEEDFINC)

Validity check routine IEEDFINC (for core storage) determines whether Main Storage Hierarchy Support is in the system. If it is not, control is passed to system reinitialization routine IEEDFINB. If it is, IEEDFINC determines whether a partition has been defined in two segments. If both H0 and H1 size have been reduced to zero, the routine marks the partition inactive in the DEFINE data area. It also checks to determine if a partition has been specified for excess bytes resulting from a redefinition in either H0 or H1 of an adjacent partition. If no partition has been specified, the routine passes control to secondary entry point IEEREXIT in command final processor routine IEEDFIN9. Otherwise, it sets up a message indicating the number of excess bytes, the partition, and the hierarchy to which they have been added. It then passes control to IEEREXIT.

If the information is valid, IEEDFINC passes control to system reinitialization routine IEEDFINB.

Listing Routine (IEEDFIN4)

Listing routine IEEDFIN4 lists partition definitions and job classes. If the time-slicing feature is in the system, it also lists the time-slicing attributes. After performing the listing function, the routine determines whether an END keyword has been read from the console, and if so, passes control to validity check routine IEEDFIN3. If not, it passes control to message routine IEEDFIN5.

Message Routine (IEEDFIN5)

Message routine IEEDFIN5 handles the messages required by the DEFINE command processing routines. These messages, which are written to the operator, are concerned with:

- Entering and continuing the definition of partitions.
- Syntax, parameter, and time-slicing errors.
- Illegal number of partitions or oversize partitions.
- Completing the definition of partitions.

After issuing the appropriate message, the routine determines whether processing is complete and if so, issues a DEFINITION COMPLETED message to all active consoles. It then determines if a DEFINITION CANCELLED message has previously been issued and if so, tests to see if the system is being initialized. If the message has been issued and it is IPL time, IEEDFIN5 passes control to command initialization routine IEEDFIN1 to repeat the DEFINE command processing. If the DEFINITION CANCELLED message has not been issued, or if it has been issued at other than IPL time, the routine returns control to the caller.

If processing is not complete, IEEDFIN5 passes control to syntax check routine IEEDFIN2.

Time-Slice Syntax Check Routine (IEEDFIN6)

Time-slice syntax check routine IEEDFIN6 checks the time-slicing entry for syntax errors. If a syntax error is found. the routine passes control to message routine IEEDFIN5 for issuance of a PARAMETER ERROR message. It ignores the erroneous entry and all following entries. If there are no syntax errors, the routine updates the DEFINE data area with the time-slicing information, gets the next entry in the statement being processed, and passes control to secondary entry point IEEDPART in syntax check routine IEEDFIN2.

Keyword Scan Routine (IEEDFIN7)

Keyword scan routine IEEDFIN7 determines whether the entry being processed is a valid keyword. If it is not a valid keyword, the routine passes control to message routine IEEDFIN5 for issuance of a PARAMETER ERROR message. It ignores the erroneous entry and all following entries. If a valid keyword is found, the routine sets the appropriate keyword indicator in the DEFINE data area.

If there are more entries to be processed, the routine gets the next entry

and passes control to secondary entry point IEEDPART in syntax check routine IEEDFIN2.

If there are no more entries to be processed (end of input), the routine determines whether a LIST keyword has been entered and if so, passes control to listing routine IEEDFIN4. If LIST was not specified, a check for the END keyword is made. If an END entry is found, the routine passes control to validity check routine IEEDFIN3. If an END entry is not found, the routine passes control to message routine IEEDFIN5 for issuance of a CONTINUE DEFINITION message.

System Reinitialization Routine 1 (IEEDFIN8)

After the partitions have quiesced, IEEDFIN8 assigns protection keys (if the system is protected) and marks dispatchable partitions not of zero size. It makes one final check to determine that no more than 15 problem program partitions have been defined. If an error is found, the routine passes control to secondary entry point IEEREXIT in command final processor routine IEEDFIN9.

If no error is found, IEEDFIN8 uses the information in the DEFINE data area to build request blocks and boundary boxes and to update the TCBPIB field and the PIB for the defined partition. The routine then passes control to IEEDFIN9 at its primary entry point, IEEDFIN9.

Before passing control to IEEDFIN9 at either entry point, IEEDFIN8 issues the DEQUEUE macro instruction specifying the boundary boxes.

Command Final Processor Routine (IEEDFIN9)

Command final processor routine IEEDFIN9 updates the time-slice control element and the task control blocks affected by time-slicing if this feature is specified.

It then tests to determine if successful partition definition has taken place. If so, it tests the CVTSMCA field of the CVT for the address of the system management control area (SMCA). If this field contains zeroes, one of two possible situations exists:

- SMF is not supported.
- SMF is supported, but has not been completely initialized at this time.

In either of these cases, or if partition definition has not completed successfully, IEEDFIN9 issues a FREEMAIN macro instruction to free the work area previously obtained by IEEDFIN3. It then passes control to IEEDFIN5 for issuance of the appropriate message specified by its caller (IEEDFIN3 or IEEDFIN8).

If the CVTSMCA field contains the address of the SMCA, SMF is supported and its initialization is complete. Partition definition has also completed successfully. Therefore, IEEDFIN9 passes control to IEEDFINA for creation of the SMF storage configuration record (type 13). Upon return from IEEDFINA, the command final processor routine passes control to IEEDFIN5 for issuance of the appropriate message.

MFT Storage Configuration Record Creation Routine (IEEDFINA)

MFT storage configuration record creation routine IEEDFINA creates the SMF storage configuration record (type 13). It receives control from SMF initialization routine IEESMF12 during SMF initialization, and from command final processor routine IEEDFIN9 whenever a DEFINE command is issued. It creates the SMF storage configuration record and issues an SVC 83 to have it transferred to the SMF buffer. It then returns control to its caller.

System Reinitialization Routine 2 (IEEDFINB)

System reinitialization routine IEEDFINB places the ECB that must be posted by the affected partition in the PIB of the partition. If a partition has been marked inactive (i.e., no H0 or H1 size is contained in the DEFINE data area), IEEDFINB sets the partition's TCB nondispatchable. If any partition being redefined contains a system writer, the routine posts the STOP ECB in the Start Parameter List to stop the writer as if a "Stop Writer" command had been issued from the console. Therefore the operator must issue a "Start Writer" command for any writer partition involved in the redefinition.

The routine then issues the WAIT macro instruction for the posting of the ECB list. After the ECB is posted, IEEDFINB issues the ENQUEUE macro instruction specifying the boundary boxes.

System Task Control (STC)

In MFT systems, system task control (See Chart 17) initiates the processing of all START commands except the START INIT command; the initiator acts as the controller and the router for all job scheduling in the system and receives control directly from SVC 34 when a START INIT command is issued.

START commands may be issued for system tasks (that operate in a zero protection key) or for problem programs (that operate in a protection key corresponding to the partition in which they are started). When the master scheduler determines that a START command with an identifier operand has been issued, it checks the validity of the partition specified in the command, builds and chains a CSCB, places a pointer to the CSCB in the partition's PIB, and posts the partition.

Note: If the START command is issued for a system-assigned reader or writer, the CSCB pointer is placed in the master scheduler resident data area.

Job selection module IEFSD510 responds when the partition is posted, and passes control to system task control whenever a START command other than a START INIT command is issued. STC processes a job description similar to a user's job description.

The job description information for a task specified in a START command comes from three sources: the procedure library, job control language (JCL) statements, and the operator. The procedure library contains standard descriptions of system tasks (the user may add descriptions of other tasks to be established via START commands). JCL statements (corresponding in input stream JCL) are created internally; these statement invoke and modify the procedure. The operator furnishes additional information in the operand of the START command; this information is edited into the internally created JCL statements before they are used to invoke and modify the procedure.

Note: Because STC does not process the START INIT command, any further references to START commands in the "System Task Control" section of this publication will exclude consideration of a START command specifying an initiator.

START COMMAND PROCESSING

System task control processes START commands issued for system tasks. System tasks are those privileged tasks such as readers and writers that operate in a zero protection key and are listed by name in linkage table IEEVLNKT.

System task control also performs processing for START commands issued for problem programs. Problem programs operate in a non-zero protection key and are not listed by name in the linkage table. In addition, data set integrity and job step timing are provided for them. Problem programs may be written by IBM with IBM-assigned names or they may be written by the user with user-assigned names. When the following conditions are met it is possible to start problem programs directly from the console via the START command:

- The JCL for the problem program has been placed in the SYS1.PROCLIB data set. (The START command specifies the name of the procedure containing the JCL.)
- The partition in which the problem program is to be started is a problem program partition.

Attribute	System Task	Problem Program Started From The Console
Protection Key	Zero	Non-zero
Name in IEEVLNKT	Yes	No
Data Set Integrity	IEEVLNKT determines this	Yes
Job Step Timing	No	Yes

Figure 10. The Differences Between System Tasks and Problem Programs Started from the Console

START Commands Issued for System Tasks

START commands are issued to schedule system tasks (see Figure 10). System task control performs the following functions prior to execution of the system task:

- Builds and interprets the JCL necessary for processing the system task.
- Constructs and initializes the control blocks and work areas necessary for execution of the system task.
- Uses the I/O device allocation subroutine to allocate the necessary I/O devices, and to provide direct access space and the mounting of appropriate volumes.
- Passes control to the system task specified in the START command.

At termination of the system task, STC:

- Uses the termination subroutine to free the I/O devices assigned to the system task, and to dispose of the data sets referred to or created during system task execution.
- Releases the main storage occupied by the control blocks and work areas.

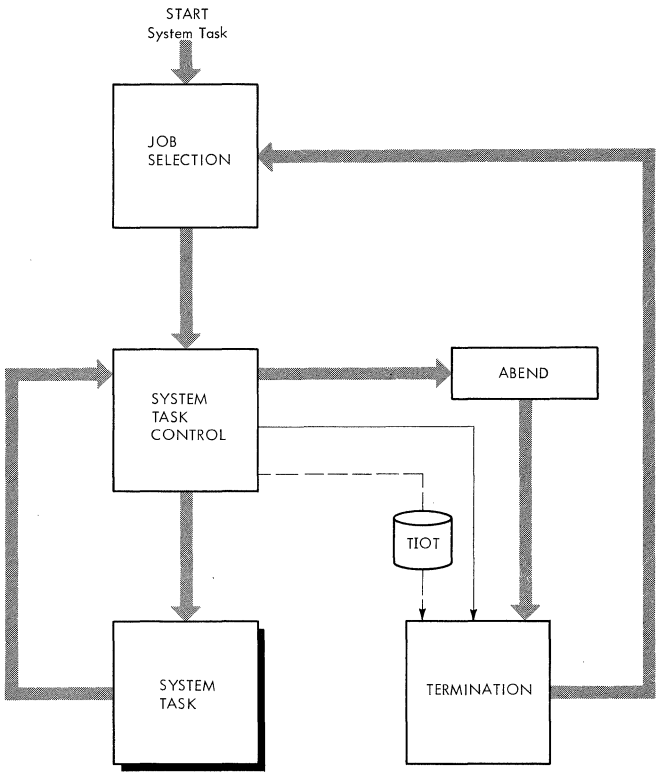


Figure 11. Scheduling a System Task in a Large Partition

START Commands Issued for Problem Programs

START commands are also issued to schedule problem programs (see Figure 12). In this case system task control:

- Builds and interprets the JCL necessary for processing the problem program.
- Performs processing necessary to bring an initiator into the partition specified in the START command.
- Passes control to the initiator.

It is important to note that the processing for problem programs started directly from the console differs from that for problem programs entered through the input stream. For problem programs entered through the input stream, it is necessary to issue a START reader command to read and interpret the necessary JCL, and a START initiator command to bring an initiator into the partition for performance of the actual processing of the problem program.

For problem programs started directly from the console, system task control performs the functions listed above, replacing those provided by the START reader and START initiator command processing routines.

Another major difference to note is that while checkpoint/restart and system management facilities are provided for problem programs entered through the input stream, they are not available for problem programs started directly from the console.

The System Task Control Routines

Seven system task control routines are common to both MFT and MVT systems. Four of these routines (IEEVSTAR, IEEVJCL, IEEVRCTL, and IEEVRC) build and interpret the JCL necessary for processing a job. Although these routines are common to both systems there are some differences in processing. Their descriptions are therefore included in this publication for the sake of clarity. The three other routines that are common to both systems are described in the MVT Job Management PLM. These routines include: internal JCL reader routine IEEVRJCL, interpreter post scan exit routine IEEPSN, and message writing routine IEEVSMSG. It should be noted that while IEEVSMSG is used in both systems, it receives control from different STC routines in the two systems.

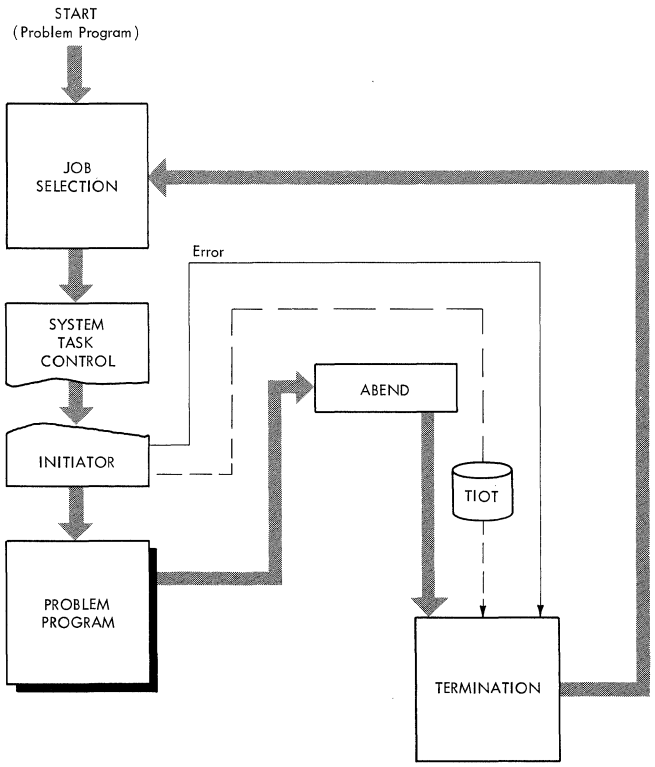


Figure 12. Scheduling a Problem Program Started from the Console in a Large Partition

LINKAGE ROUTINES (IEFSD586, IEFSD587, IEFSD588, IEFSD589, IEF589SP) AND LPSW ROUTINES (IEFSD534, IEFSD535, IEFSD584, IEFSD585): In an MFT system, the initiator acts as the controller and the router for all job scheduling in the system. The initiator routines are therefore required to be in the supervisor state so that they may perform privileged operations. The system task control routines, however, do not perform privileged operations and therefore operate in the problem program state. Thus, it is necessary to maintain an interface between the initiator and system task control.

This interface is accomplished by a series of linkage routines and LPSW routines. A linkage routine, operating in the supervisor state, receives control from the initiator and passes control to a Load PSW routine via a LINK macro instruction. Therefore, when control is eventually returned to the linkage routine, the supervisor state of the linkage routine is maintained. The linkage routine is then able to return control to the initiator via an XCTL macro instruction, preserving the supervisor state of the initiator. (See Chart 17.)

When a Load PSW routine receives control from a linkage routine, it issues a Load PSW instruction for entering the problem program state. All subsequent STC routines therefore operate in the problem program state.

While different linkage routines and LPSW routines are used, depending on the STC processing required, their functions are all similar. One exception should be noted, however. Linkage routine IEFSD588 does not pass control to a Load PSW routine. Instead, it passes control directly to transient reader linker routine IEF591SD via a LINK macro instruction. In this case, IEF591SD issues the LPSW instruction to enter the problem program state for system task control.

START SYNTAX CHECK ROUTINE (IEEVSTAR): The START syntax check routine gets main storage for, and builds, the start descriptor table (SDT) (see Figure 13). Seven entries are provided in the SDT: the first contains the JOB statement, the second contains the EXEC statement that calls the procedure specified in the START command, the remaining entries are provided for a DD statement and continuations of the EXEC and DD statements. Each entry contains a one-byte identification flags field, whose bits, when set to one, have the following meanings:

- Bit 0 indicates a JOB statement.
- Bit 1 indicates an EXEC statement.
- Bit 2 indicates a DD statement.
- Bit 3 indicates a DD statement continuation.
- Bit 4 indicates an EXEC statement continuation.
- Bits 5 through 7 are reserved.

The routine generates the JOB, EXEC, and DD statements that are placed in the SDT. The keyword parameters in the START command are compared with a list of keyword parameters that are allowable in a DD statement; they are not compared with DD subparameters. If the keyword corresponds to a member of the list, the routine stores it in the DD statement in the SDT. This DD statement overrides the IEFRDER DD statement in the procedure specified in the START command. If the keyword does not correspond to a member of the list, it is assumed to be a symbolic parameter keyword and is placed in the EXEC statement in the SDT. The routine then constructs the required allocate parameter list, and a JSCB and WTPCB in subpool 255 (SQA). It then stores the JSCB address in the TCB and CSCB.

Finally, the syntax check routine passes control to JCL edit routine IEEVJCL.

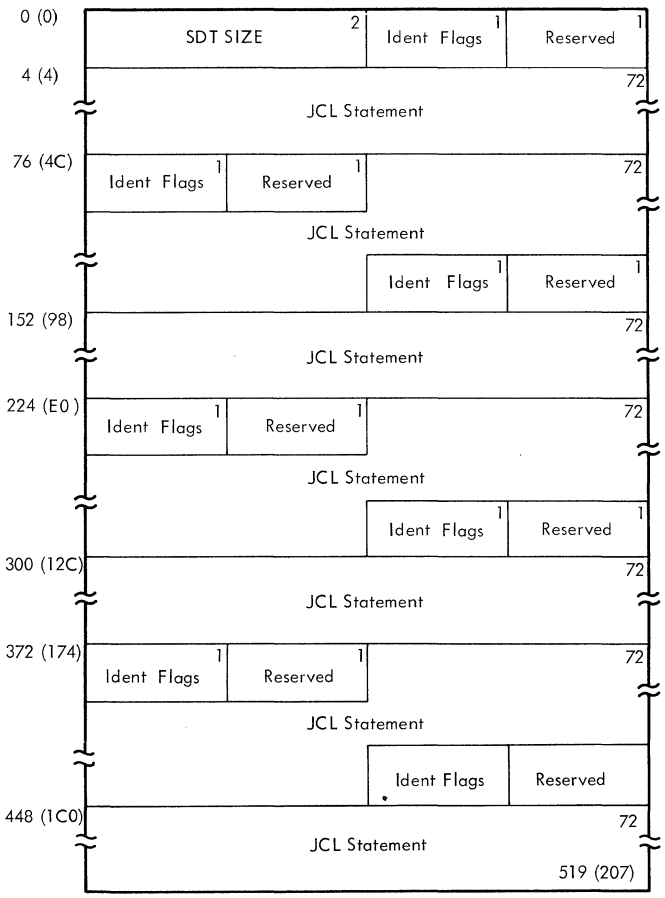


Figure 13. START Descriptor Table (SDT)

JCL EDIT ROUTINE (IEEVJCL): JCL edit routine IEEVJCL builds the job control language set (JCLS). Using the information in the SDT, the JCL Edit routine puts the JCL in the form appropriate for the interpreter. Each statement is built in an 88-character buffer (obtained with a GETMAIN macro instruction). A pointer to the first buffer is placed in the CSCB associated with the START command. Each buffer contains a pointer to the next buffer, 4 bytes of reserved space, and a "card image" of the statement in the last 80 bytes. The routine then builds the following:

- The Job Scheduling Entrance List (JSEL).
- The Job Scheduling Exit List (JSXL).
- The Job Scheduling Option List (JSOL).
- The Option Buffer (OPT).

READER/INTERPRETER CONTROL ROUTINE (IEEVRCTL): Reader/interpreter control routine IEEVRCTL receives control and builds the interpreter entrance list (NEL), option list, and exit list, and the Job Scheduling Work Area (JSWA). interpreter entrance list contains the address of the JCLS in its third word. The routine also issues a LOAD macro instruction specifying internal JCL reader routine IEEVRJCL and interpreter post scan exit routine IEEPSN so that these modules may be used by the reader. IEEVRCTL then creates entries in the NEL exit list for these modules and for interpreter exit routine IEEVRC. The reader/interpreter control routine then frees main storage used for the JSOL and the OPT and passes control to the reader via an XCTL macro instruction.

The nonzero value of the third word of the entrance list indicates that the input stream is an internal data set. Since the input stream is internal, the reader issues a pseudo OPEN macro instruction to bring a special access method (a modified QSAM) into storage and places a pointer to the access method in the input DCB. This special access method reads the JCLS; it is entered from the expansion of the standard GET macro instruction.

The internally-stored job control language statements, and the statements from the procedure library are analyzed and combined. The standard job description tables are built, and an input queue entry is constructed; however, because bit 7 of the option switches field of the option list is off, the entry is not enqueued, and the reader or writer "job" cannot be selected by an initiator. If errors are detected during reader processing, appropriate messages are placed in system message blocks, which are enqueued in the message class queue. When processing is complete, the reader places the main storage address of the job control table (JCT) in the NEL and passes control to interpreter exit routine IEEVRC via an XCTL macro instruction specifying the exit list entry for IEEVRC created by the reader/interpreter control routine.

INTERPRETER EXIT ROUTINE (IEEVRC): When interpreter exit routine IEEVRC receives control, it issues a DELETE macro instruction specifying the internal JCL reader routine and the interpreter post scan exit routine. Routine IEEVRC writes the logical track header (LTH) onto the job queue and uses the Queue Management routines IEFCNVRT and IEFRDWRT to indicate that, for warm start, the job should be processed as if it had been dequeued from an input queue. It analyzes the return codes from the reader and the scan routine and performs the appropriate processing. For internal JCL errors and for I/O errors, it initializes the CHSPA field of the CSCB with an appropriate error code. (These error codes are explained in the description of the CSCB found in Appendix A.) If a system task is being started and either of these types of errors is detected, the interpreter exit routine sets the job-failed bit in the JCT to terminate the starting task. For all other errors, IEEVRC also sets the job-failed bit in the JCT to terminate the starting task.

The interpreter exit routine then frees main storage for the JCLS, the reader/interpreter register save area, the NEL, the options list, and the exit list. It also frees main storage for control blocks used by MVT systems (i.e., JSWA, JSEL, and JSXL).

If the START command was issued for a system task, the routine then passes control to allocation interface control routine IEEVACTL. If the START command was issued for a problem program, the interpreter exit routine determines if the partition specified in the START command is a problem program partition. If not, the routine sets the job-failed bit in the JCT to terminate the starting task, and issues an appropriate message to the operator. It then passes control to the allocation interface control routine.

If the START command was issued for a problem program specifying a problem program partition, IEEVRC returns control to linkage module IEFSD589 with a return code of four in register 15.

ALLOCATION INTERFACE CONTROL ROUTINE (IEEVACTL): The allocation interface control routine sets up the necessary parameter list for IEEVSMSG to issue the WTO macro instruction to inform the

operator of any errors that have been found. Finally it passes control to the I/O device allocation routine via a LINK macro instruction.

I/O device allocation routine IEFSD21Q uses the JCT to find the appropriate tables in the input queue, allocates the necessary devices to the reader or writer, and issues any necessary mounting messages. The allocation recovery routines issue WTO macro instructions to inform the operator of any errors found during allocation. When allocation is complete, or if allocation cannot be performed, control is returned to the allocation control interface routine.

Allocation control interface routine IEEVACTL determines if the routine to be given control is an authorized routine and then transfers control to Write TIOT routine IEESD590.

Note: The linkage table contains a list of all "authorized" routines. The module name of this table is IEEVLNKT. It is used by both MFT and MVT systems and its format may be found in the "Command Processing" section of the MVT Job Management PLM.

QMPA BUILDER ROUTINE (IEEVSMBA): The QMPA builder routine obtains main storage for and builds:

- The message class queue manager parameter area (QMPA).
- Its associated ECB/IOB.

It uses the Queue Management Read/Write Routine to read in the SYSOUT class directory (SCD) for the task from the job queue. It uses information in the SCD to initialize the QMPA. IEEVSMBA then builds the ECB/IOB for the QMPA and stores the address of the QMPA in the LCT.

The QMPA builder routine then returns control to its caller.

WRITE TIOT ON DISK ROUTINE (IEESD590): Write TIOT on disk routine IEESD590 writes the TIOT and checks to see if a system task is to be started in a small partition. If not, it passes control to linker routine IEESD591 via an XCTL macro instruction. If so, the routine posts the "no work" ECB in the PIB. It then posts the ECBA in the SPIL with a completion code of zero to indicate to small partition routine IEFSD599 that initiation services have been performed and that the system task is ready to be executed. The routine then waits on the ECBC in the SPIL. When the ECBC is posted by the small partition routine, IEESD590 returns control to linkage routine IEFSD589.

LINKER ROUTINE (IEESD591): The linker routine (load module IEESD591) consists of two assembly modules: IEESD591 and IEE591SD. The routine is entered at IEESD591 from write TIOT routine IEESD590. It passes control to the system task specified in the START command via a LINK macro instruction. When the task stops or suspends processing, control is returned to the linker routine. If an I/O error occurred, IEESD591 issues the ABEND SVC. The routine then returns control to the appropriate calling routine.

The routine is entered at IEE591SD from linkage routine IEFSD588 to restore a transient reader. It issues a Load PSW instruction to enter the problem program state. It then passes control to the transient reader via a LINK macro instruction. When the transient reader stops or suspends processing, control is returned to IEE591SD, which returns control to linkage routine IEFSD588.

TERMINATION INTERFACE CONTROL ROUTINE (IEEVTCTL): The termination interface control routine sets up the necessary tables and parameters for termination processing and passes control to termination entry routine IEFSD42Q via a LINK macro instruction. Upon return from termination, it frees main storage used for the various tables and passes control to POST routine IEESD592 via an XCTL macro instruction.

MESSAGE WRITING ROUTINE (IEEVOMSG): Message writing routine IEEVOMSG assembles and writes messages to the operator for termination interface control routine IEEVTCTL.

POST ROUTINE (IEESD592): POST routine IEESD592 posts the "no work" ECB in the PIB. If termination services have been performed for a small partition, the POST routine also posts the ECBA in the SPIL with a completion code of two to indicate that job termination has taken place. If a CSCB address has been passed to IEESD592, the routine removes the CSCB from the CSCB chain and frees the main storage that it occupied. Finally, the POST routine adjusts the boundary box contents to indicate that the partition is now cleared for further processing, and returns control to the appropriate calling routine.

PART 4: COMMON ELEMENTS OF JOB MANAGEMENT

Job management routines frequently refer to or execute system elements that are also used by other job management routines. The common elements of job management are discussed below.

- The work queues are the temporary storage areas that permit work to be stored in input sequence, and to be processed in priority sequence. They act as buffers between the interpreters, initiators, and system output writers, allowing each to process at maximum speed.

- The system management facility is an optional feature of the control program that provides a means for gathering and recording the type of information that can be used to evaluate system usage.

- The write-to-programmer facility enables the operating system to provide the user with information about his job in the event of an abnormal occurrence during execution. This information is conveyed to the user via messages to the system message class output data set.

Work Queues

MFT and MVT systems use the same job queue data set (SYS1.SYSJOBQE). It consists of work queues that occupy space on a permanently resident volume. The space for the job queue data set is allocated when the system is generated in response to a DD statement submitted by the installation. The DD statement specifies the amount of space (contiguous tracks) to be allocated.

The work queues forming the job queue data set consist of:

- The automatic SYSIN batching (ASB) queue.
- The HOLD queue.
- The remote job entry (RJE) queue.
- 36 output class queues.
- 15 input job class queues.
- The background reader (BRDR) queue.
- 20 unused queues.

In MFT systems the background reader queue is also unused.

The job entries are enqueued in priority order within each job class on the appropriate job class queue. Jobs are selected for processing according to the job class designation of the partition requesting work.

QUEUE MANAGEMENT

Queue Management is a general term describing a group of routines used by various system components, such as the reader/interpreter, initiator/terminator, and output writer. The queue manager performs some common functions for all system components. It performs all input/output for accessing the job queue data set and keeps track of all space on this queue. The queue manager assigns space on the job queue in logical track increments for control blocks, tables, and system messages built by the scheduler. When the control blocks and tables have been created, the reader/interpreter enqueues (ENQs) the job using the queue manager. After the job is enqueued, the initiator dequeues (DEQs) the job for execution when a partition that is assigned to service that job class becomes available for work. The terminator places control information needed by the system output writer on the job queue. At job termination, the terminator enqueues the output work description. The writer then dequeues the output work according to output class and priority within the class, and transcribes it to the appropriate device, specified by the user.

At system generation, the space for the job queue data set is allocated. The device upon which the job queue resides is considered a non-demountable system residence volume.

MFT uses the MVT Queue Manager. However, to reduce possible interlocks due to unavailability of requested tracks, the assign routine (IEFQASGQ) has been modified, and another module (IEFSD572) has been added. The discussion of the queue manager includes descriptions of some MVT modules to provide a more complete explanation of the relationship of these modules to the entire system. A discussion of the transient queue manager (SVC 90) is also included.

JOB QUEUE INITIALIZATION

At system initialization, queue initialization routine IEFSD055 receives control from the SET command processor to construct a data control block (DCB) in the nucleus, and to issue an OPEN macro instruction which causes a data extent block (DEB) to be built for accessing SYS1.SYSJOBQE. It also places a queue manager master queue control record (master QCR) in the nucleus after the DCB and DEB. The master QCR contains information about the queue data set as a whole, and is used in the control and maintenance of the free-track queue. (See Figure 14 for the format of the master QCR.) Control then passes to queue formatting routine IEFORMAT.

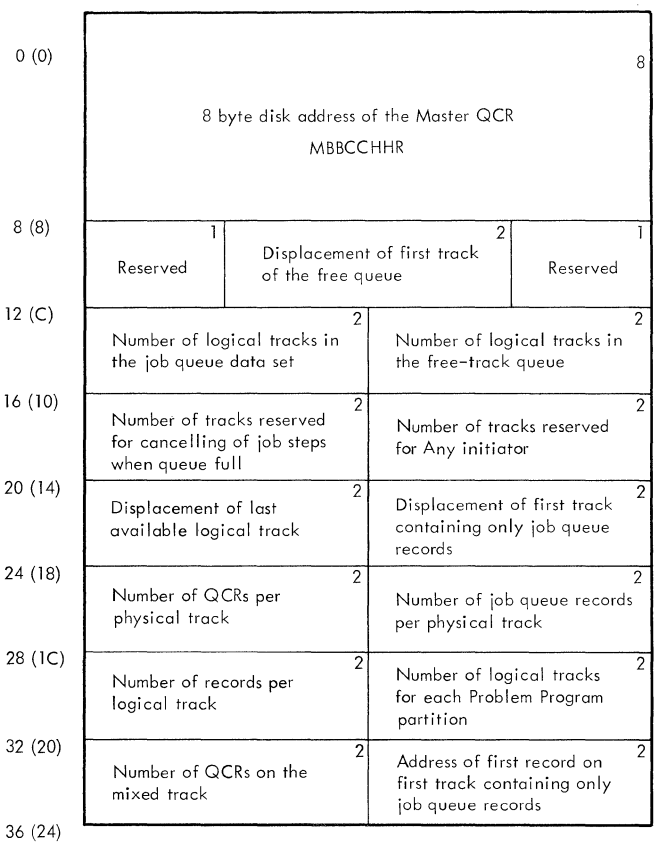


Figure 14. Master Queue Control Record (Master QCR) Format

The queue formatting routine divides the job queue data set into a control record area and a logical track area. The control record area contains a copy of the master QCR, and a QCR for each of the remaining queues. Each QCR is used in the maintenance of a single queue; there is one ASB queue control record, one HOLD queue control record, one RJE queue control record, 36 output queue control records, 15 input queue control records, one BRDR queue control record, and a queue control record for each of the 20 unused queues. (See Figure 15 for the format of an input queue control record.)

Note: The first position of the job queue control record (job QCR) contains zeros if no work exists. The job QCR contains a minimum of two entries if work exists for at least one priority.

The job class specified by the user (on the JOB statement or in a START command) is converted by the system to match the system-assigned job class identifiers. The user-assigned job class and corresponding system job class identifiers are:

User-Assigned Job Class	System-Assigned Identifier (Hexadecimal)
A	28
B	29
C	2A
D	2B
E	2C
F	2D
G	2E
H	2F
I	30
J	31
K	32
L	33
M	34
N	35
O	36

The logical track area length is variable. Logical tracks are used instead of physical tracks so that the job queue can reside on different device types. Each logical track contains a 20-byte header record (LTH) (as shown in Figure 16) which includes a pointer to the next track. The header record is used to chain all tracks of a job together. When the job is enqueued, the header record is used to chain jobs first-in/first-out (FIFO) according to priority. All jobs of the same job class are chained together. Following the header record are a variable number of 176-byte data records. The number of records per logical track is determined at system generation and may range from 10 to 255 records. The number may be modified within this range at IPL. All tables, control blocks, and system messages are in 176-byte increments.

At system initialization, all tracks are members of the free track queue. The free track queue is a list of logical tracks

available for assignment to work queues. As tracks are needed, they are taken from the free track queue. When the system is finished with tracks, they are returned to the free track queue. After system initialization, SYS1.SYSJOBQE appears as shown in Figure 17. Figure 18 illustrates typical input and output work queues. Each input and output QCR contains the address of the last entry in each priority queue.

QUEUE MANAGER MODULES

As jobs are read into the system, they are placed into each job class queue according to priority (established by the PRTY parameter on the JOB statement). When the reader/interpreter reads a job or establishes a new queue for an output class, it establishes a queue entry. This is done by Assign/Start Routine IEFQASGT.

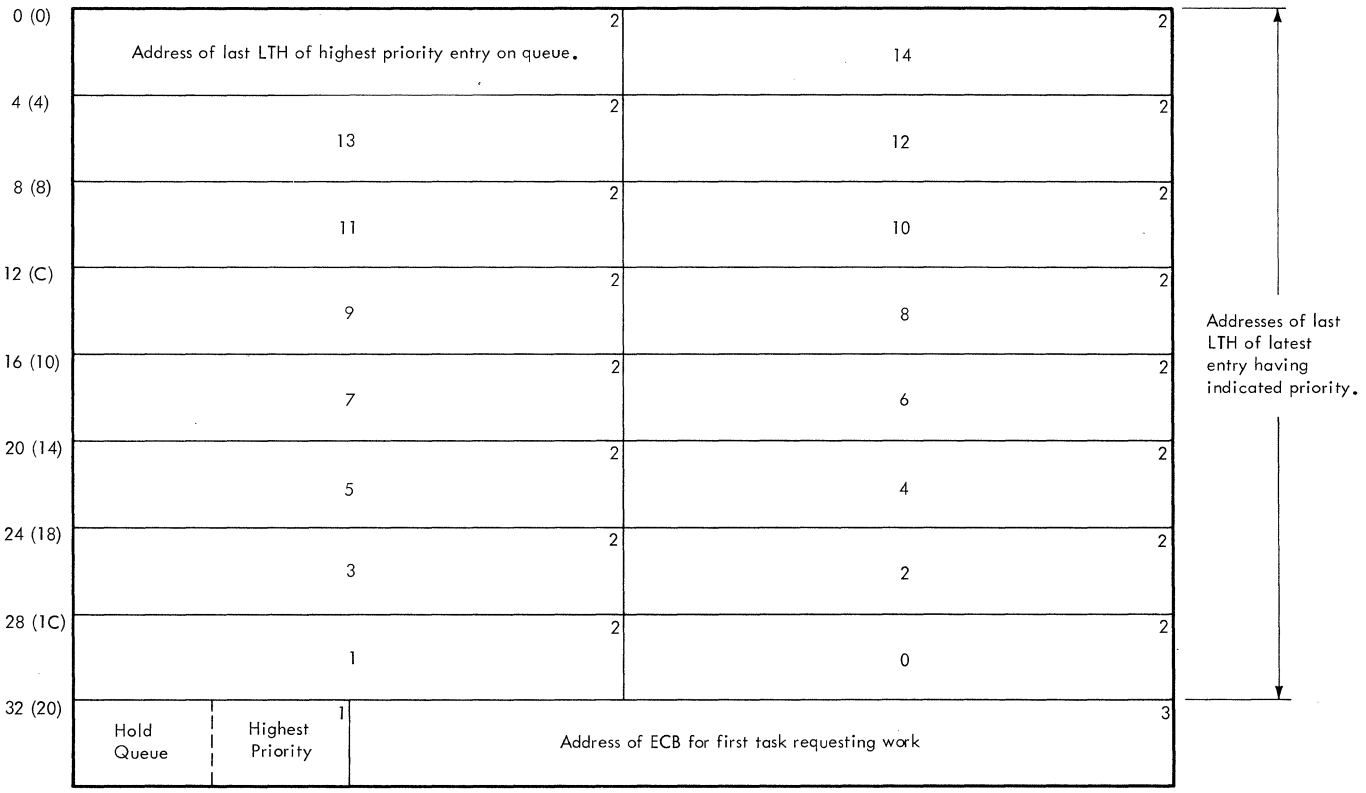
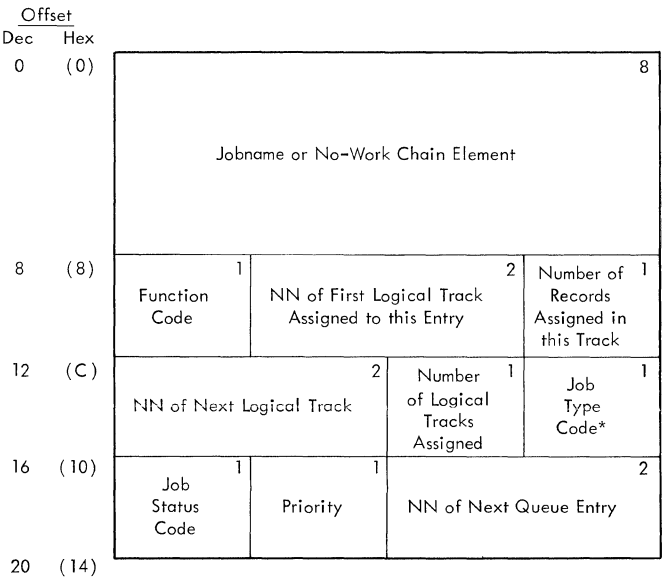


Figure 15. Job Queue Control Record (QCR)



* Job Type Code
1 = HOLD queue
2 = ASB queue
3-38 = Output class queues
39 = RJE queue
40-54 = Input work queues
55 = BRDR queue

Figure 16. Logical Track Header (LTH) Record Format

Assign/Start Routine (IEFQAGST)

The Assign/Start routine takes the first track from the available track pool and establishes it as the first track for a job. The queue manager parameter area (QMPA) is updated accordingly. (See the MVT Job Management PLM for a description of QMPA.) An IOB and an ECB are created for subsequent input/output operations. The actual reserving of tracks is done by the assign routine, IEFQASGQ.

Note: MFT does not support the track-stacking facility of MVT.

Assign Routine (IEFQASGQ)

The assign routine assigns record space on the job queue, and determines whether the requested blocks can be assigned to the current track. If so, the record addresses are placed in the external parameter list of the QMPA, and the records-available field of the QMPA is decremented to reflect this assignment. If additional logical tracks must be assigned, this routine issues an ENQ macro instruction on the master QCR to prevent concurrent access by other tasks. The master QCR is read into main storage.

The primary user of this assign routine is the reader/interpreter, although the initiator/terminator also uses it. To prevent the possibility of the reader/interpreter taking all the space and making it impossible for jobs to be initiated or terminated, two limit values have been added: the number of tracks reserved for initiating a job, and the number of tracks reserved for terminating a job.

If logical tracks are available, the requested tracks are acquired. The address of the first available logical track is updated and the newly assigned tracks are chained to the tracks assigned to the job. The master QCR is written to the control record area of the job queue data set. A DEQ macro instruction is issued to make the master QCR available to the next user.

If there are no available logical tracks, and the requesting routine is a reader/interpreter, the assign routine passes control to queue manager/interpreter interlock routine IEFSD572. If the reader/interpreter is resident, control returns to the assign routine to wait for tracks to become available. If the reader/interpreter is transient, IEFSD572 issues a message to the operator requesting him to reply "WAIT" or "CANCEL". If the reply is WAIT, control returns to the assign routine, otherwise control is passed to the ABEND routines to cancel the reader/interpreter.

If there are no available logical tracks and the requesting routine is an initiator/terminator, the assign routine issues a message to the operator stating that queue space has been exceeded and passes control back to the initiator/terminator to cancel the job.

When the requesting routine is assigned the record TTRs, it can read and write records on the job queue. The master QCR is written, and a DEQ macro instruction is issued to make the master QCR available to the next user. The record addresses in storage and TTR pointers are contained in the external parameter list of the QMPA. When available space on the job queue becomes critical, a warning is sent to the requesting task. Logical tracks are removed from the pool of available tracks and assigned to the job.

If the reply is CANCEL, the interlock routine deletes all queue space assigned to the job, cancels the job, and returns control to the assign routine. Normal initiator operation recovers the partition for further use.

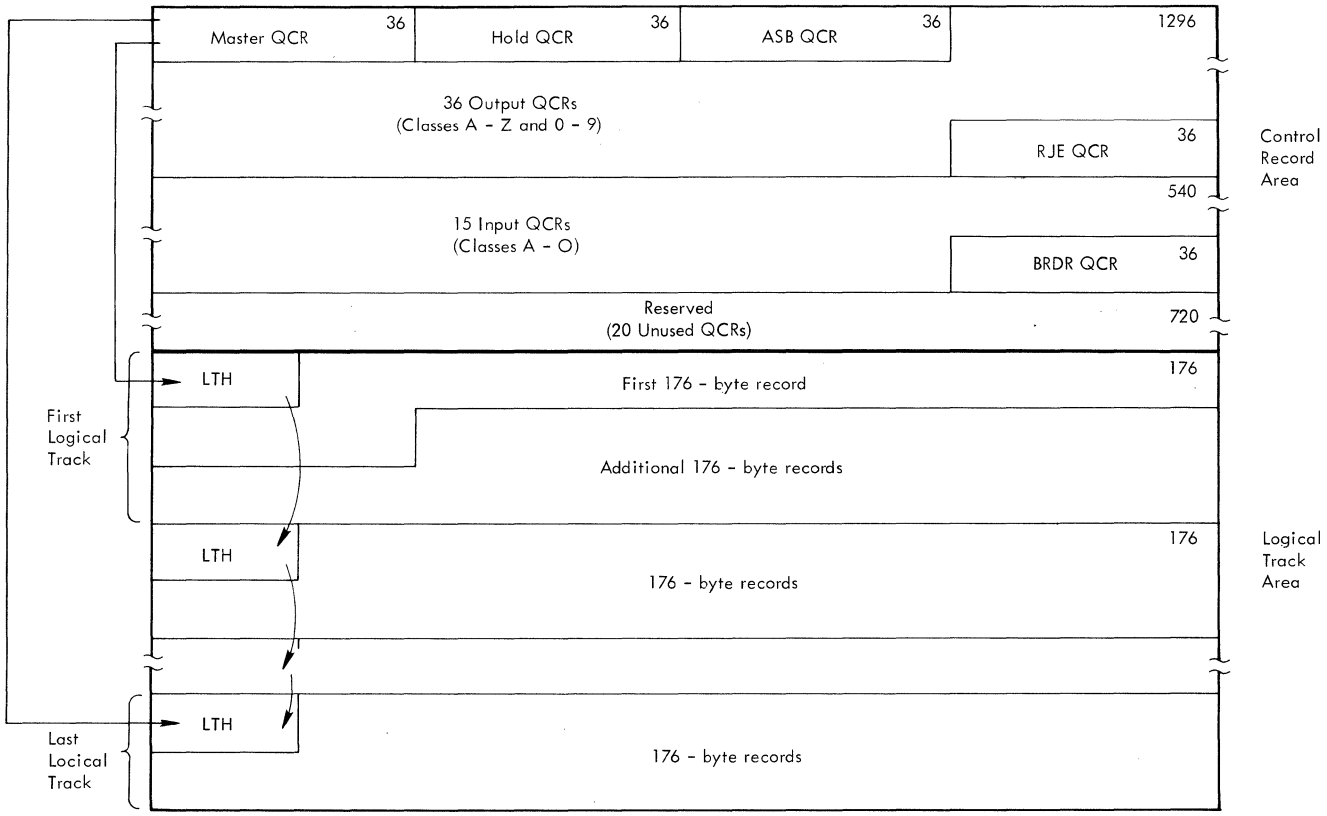


Figure 17. Sample Job Queue (SYS1.SYSJOBQE) Format After Initialization

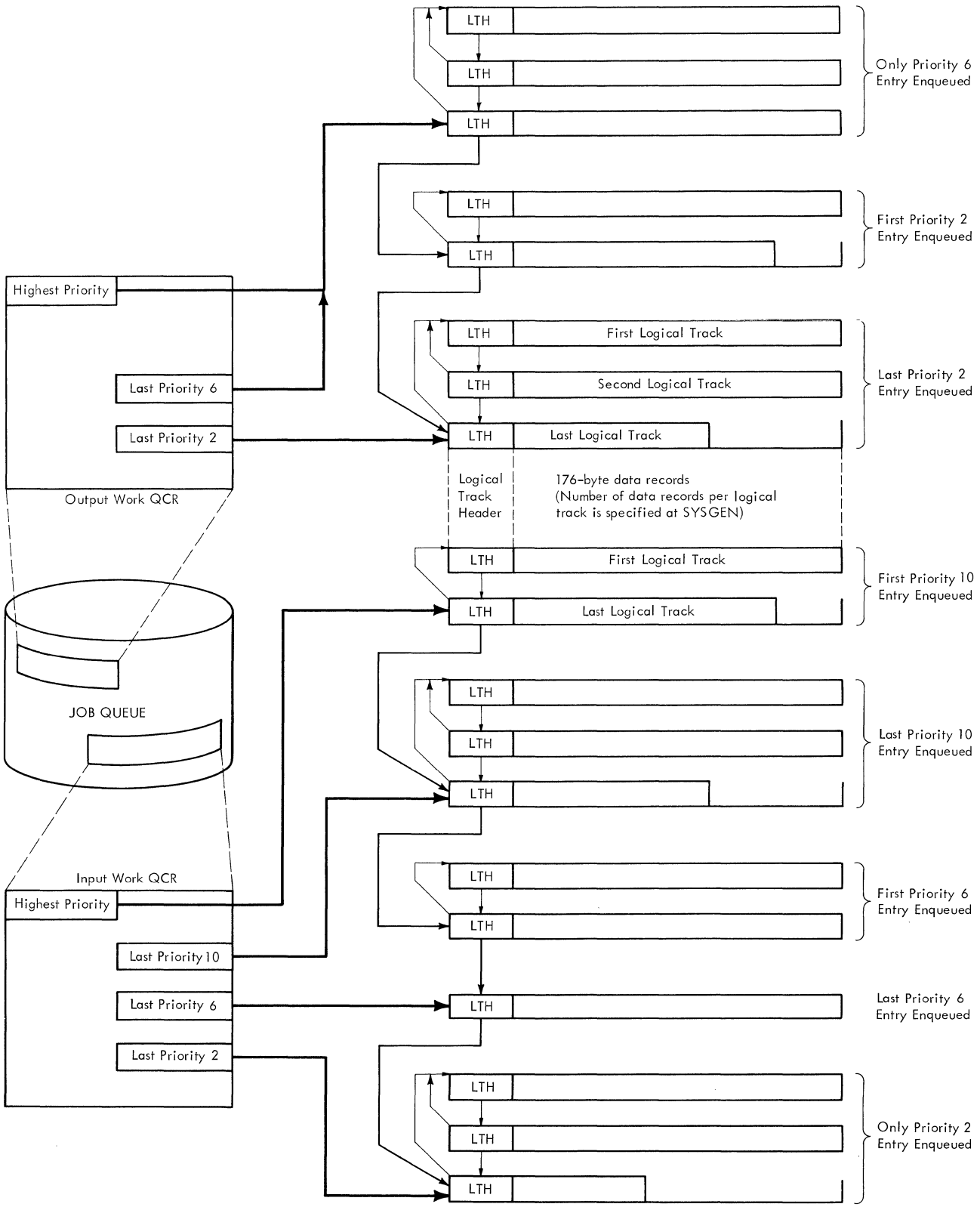


Figure 18. Input and Output Queue Entries

Interpreter/Queue Manager Interlock Routine (IEFSD572)

When the reader/interpreter requests tracks for the job it is processing, and no space is available, IEFQASGQ passes control to interlock routine IEFSD572 to identify whether an interlock can occur. If the reader is transient, the possibility exists that space needed by the reader/interpreter can be provided only by the termination routines, which must operate in the partition that the reader occupies. Because the requested space is not available, the routine issues a message to the operator requesting a reply of 'WAIT' or 'CANCEL'. If the reply is WAIT, this routine returns to the assign routine to wait for available space. (If the reader requesting space is a resident reader, no message is issued, and a reply of WAIT is assumed.)

If the reply is CANCEL, control passes to delete routine IEFQDELQ to delete all queue space assigned to the job being processed (if any space had already been assigned). When control returns, the interlock routine abnormally terminates the job with a job-canceled code of 222. Normal initiator operation recovers the partition for further use.

Queue Manager Enqueue Routine (IEFQMNQQ)

After all control blocks for a job have been written, the job is eligible for selection by an Initiator. Declaring a job ready for selection (enqueuing) is done by Queue Manager Enqueue routine IEFQMNQQ.

When an interpreter has completed the processing of a job, (all records generated by the interpreter have been written on the queue), it uses this routine to enqueue the job, in priority order, on the appropriate job class input work queue. When a job completes processing, the terminator uses this routine to enqueue output data sets, in priority order, on the appropriate output work queues.

To prevent concurrent updates, this routine issues an ENQ macro instruction for the queue control record (QCR) of the proper queue. When the QCR becomes available, it is read into main storage. The enqueue routine then places the new queue entry after the last entry with the same priority as shown in Figure 16. The address of the new entry is then placed in the track header of the prior entry (maintaining a chain), and in the QCR position for that priority. The job control table (JCT) is written. The updated QCR is written on the job queue. A DEQ macro instruction is issued making the QCR available. Control is then returned to the calling routine.

Dequeue Routine (IEFQMDQQ)

In addition to dequeuing a job from the input queue for an initiator, the dequeue routine (IEFQMDQQ) removes the output data from an output queue for processing by a system output writer.

The routine issues an ENQ macro instruction on the QCR of the selected queue. When the QCR becomes available, the dequeue routine reads it into main storage. The QCR is examined for a job belonging to the same job class as the partition. Upon finding a job, this routine adjusts the chain. If none is found, the requesting task tries the next job class. If no work is found on any of the selected queues (up to three), the requester places itself in a wait state. In the case of an output writer, a pointer to the "no work" ECB is placed in the QCR. If a pointer already exists, the ECB is chained to the last ECB waiting for that output class. Then the updated QCR is written and a DEQ macro instruction is issued making the QCR available.

Once a job has completed processing, or the output writer has written all records for a job, the tracks are returned to the system. This is known as deleting a job and is handled by the queue manager delete routine IEFQDELQ.

Delete Routine (IEFQDELQ)

The Delete routine first issues an ENQ macro instruction on the master QCR of the free chain of tracks. After control is returned, the record is updated to reflect the new available tracks. The prior last track of free storage is updated to point to the new set of free tracks. After the master QCR is updated, it is written and a DEQ macro instruction is issued against it. The ECB indicating wait-for-space is posted.

Figure Breakup Routine (IEFSD514)

When a reader must be suspended, the job scheduler must prevent the destruction of variable size tables in main storage. To do this, it calls the queue manager table breakup routine, IEFSD514, (Chart 4) which subdivides tables in main storage and writes them on disk as 176-byte data records. The data records are written in a queue entry related to the caller. The job scheduler calls IEFSD514 to retrieve the

176-byte data records and to reconstruct the tables in main storage. Whether reading or writing tables, the caller must build a parameter list (see Figure 19) and place the address of the list in general register 1 before calling the TBR.

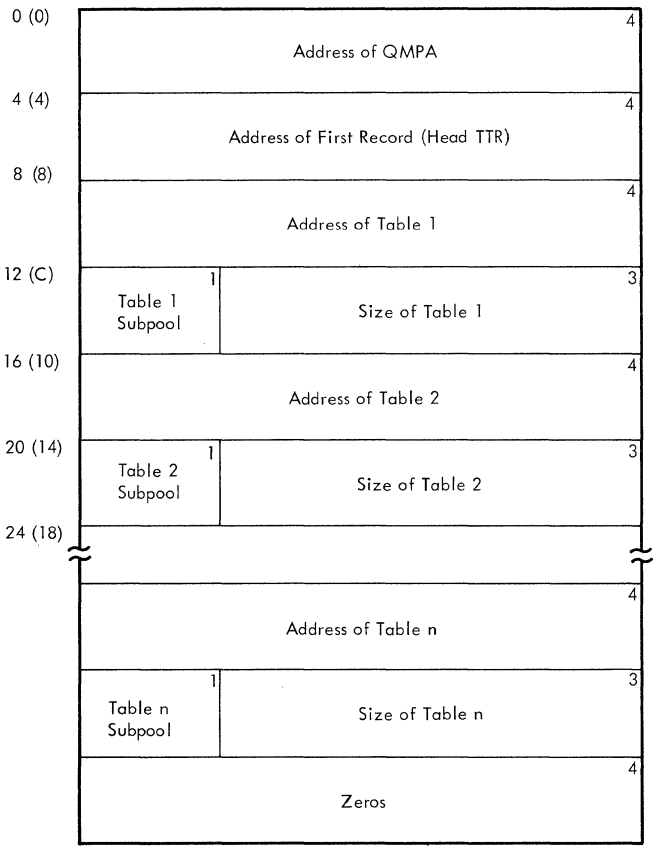


Figure 19. Table Breakup Parameter List

When the tables are written initially, the TBR parameter list must contain the address of a QMPA specifying the queue entry into which the tables are to be written. The function code field (QMPOP) of QMPA must specify a write operation. The TBR parameter list must also contain the address, subpool, and size of each table to be written. The last word of the TBR parameter list must be zero. The TBR returns a Head TTR address which locates the beginning of the tables on disk. This TTR must be saved for subsequent retrieval of the tables.

The initial write establishes disk data records for the tables for the duration of the associated queue entry (i.e., until the entry is deleted). Therefore, further write requests must specify the Head TTR in the TBR parameter list. Before issuing a write request, the caller must retrieve any previously written tables to prevent their being overlaid by the new write request.

If the request is for output of tables, (transferring from main storage to direct access device), the Head TTR (passed in the parameter list) is used to read the first table queue control record (TQCR). If the Head TTR is zero, the assign routine, IEFQASGQ, is called to assign space for a new TQCR. The TQCR is a 176-byte record containing a 4-byte forward-chain pointer and space for 43 TTRs. These spaces are filled in as the tables are written, using the assign routine to assign the TTRs, and the Read/Write routine, IEFQMRAW, to write the tables in 176-byte segments. If more than 43 records are required to hold the tables, a new TQCR is chained to the first, and processing continues. The low-order byte of the last TTR used in writing the tables is set to 'FF' (hexadecimal) to indicate end-of-tables. After these TTRs are assigned, they are used each time the table breakup routine is called to write tables, as long as the Head TTR is preserved by the caller.

Once a queue entry has been deleted, a caller must issue another initial write request (Head TTR is zero in the table breakup routine parameter list) to establish a new string of table data records. IEFSD514 does not free table storage areas.

In retrieving tables, the TBR parameter list must contain the address of an associated QMPA. The function code (QMPOP) field must specify a read operation. The TBR parameter list must also contain the Head TTR address. Sufficient space must be allowed for the TBR to return the new main storage address of each table, and the subpool and size of each table as specified when they were written by the TBR.

If the request is for input (reading into storage) of tables, the first TQCR is read into storage using the Head TTR passed in the parameter list. The first record of the first table is read, using the first record in the TQCR. This record contains the size of the table and the number of the desired subpool. IEFSD514 issues a GETMAIN specifying the subpool and the amount of storage required for the table. The remainder of the table is then read into the storage obtained, using read/write routine IEFQMRAW. Each table specified in the parameter list is processed in this manner until 'FF' (hexadecimal), indicating end-of-tables, is found. As each table is read into main storage, the parameter list is updated with the main storage address of that table. When all tables have been read, control is returned to the caller. The address of the updated parameter list is returned in register 1. Figures are always written in the same sequence that they appear in the TBR parameter list,

beginning with the Head TTR. They are retrieved, totally, in the same sequence; they cannot be read selectively.

Transient Queue Manager Routines (IEFXQM00, IEFXQM01, and IEFXQM02)

The transient queue manager consists of initialization and read/write routine IEFXQM00, track assignment routine IEFXQM01, and record assignment routine IEFXQM02. These routines provide the services of assign/start routine IEFQAGST, assign routine IEFQASGQ, and read/write routine IEFQMRAW. The transient queue manager is in SYS1.SVCLIB and operates in the transient SVC area.

When the transient queue manager initialization and read/write routine IEFXQM00 receives control it first initializes an ECB/IOB and prepares the QMPA. If the queue manager was requested to provide a track or record, IEFXQM00 branches via an XCTL macro instruction to track assignment routine IEFXQM01 or record assignment routine IEFXQM02. If the queue manager was requested to read or write a record onto the job queue, IEFXQM00 performs the read or write. IEFXQM00 returns control to the caller upon completion, as does IEFXQM01 and IEFXQM02.

System Management Facility

The system management facility is an optional feature of the control program that provides a means for gathering and recording the type of information that can be used to evaluate system usage. It consists of routines that collect data, routines that record the collected data in a data set, and routines that provide interfaces for exits to user-supplied data collection routines. The system management facility may include either one tape data set (SYS1.MANX) or two direct access data sets (SYS1.MANX and SYS1.MANY). In the latter case, the two data sets are filled alternately: while one is in use, the other may be dumped via the SMF dump routine (IFASMFDP).

SMF data collection routines and exits for user-supplied data collection routines are in the Supervisor, the Interpreter, the Initiator, and the Termination routines. IBM-supplied data collection routines are in the Supervisor, the System Output Writer and the Vary Command Processor.

The routines that record the data in the SMF data set include the SMF SVC routines (SVC 83) and the SMF writer routine (IEESMFWT).

COMPARISON OF SMF IN MFT AND MVT

The areas of SMF processing that are the same in MFT and MVT include:

- The SMF SVC routines (SVC 83).
- The SMF dump routine.
- The SMF processing in the Interpreter, the System Output Writer, the VARY Command Processor, and the Termination routines.

This processing is described in the MVT Job Management PLM.

Additional SMF processing in MFT job management routines differs from MVT only in the manner in which it is implemented. The areas of difference in implementation occur in:

- The SMF initialization procedure.
- The SMF writer routine.
- The SMF processing in the Initiator.

SMF processing in the Initiator is described in the "Job Processing" section of this publication. The SMF writer routine and the processing for SMF initialization is described below.

The SMF records are the same for MFT and MVT, except for the addition of an SMF storage configuration record (type 13) in MFT. A description of the SMF records, showing the data elements that occur in each record type and the source from which each data element is obtained, is found in the "Common Elements of Job Management" section for the MVT Job Management PLM.

SMF Initialization

In MFT the routines initializing the system management facility operate under the SMF TCB and the master scheduler TCB. (See Figure 20.) SMF writer routine IEESMFWT executes under the SMF TCB. It is initially involved in a WAIT/POST interchange with the SMF initialization routines operating under the master scheduler TCB. (This is to ensure that the system management control area (SMCA) is initialized and that IEESMFWT has access to it before the writer routine begins initialization processing.) The SMF writer routine's primary initialization function is to issue an SVC 83 for the opening and allocation of the SMF data sets.

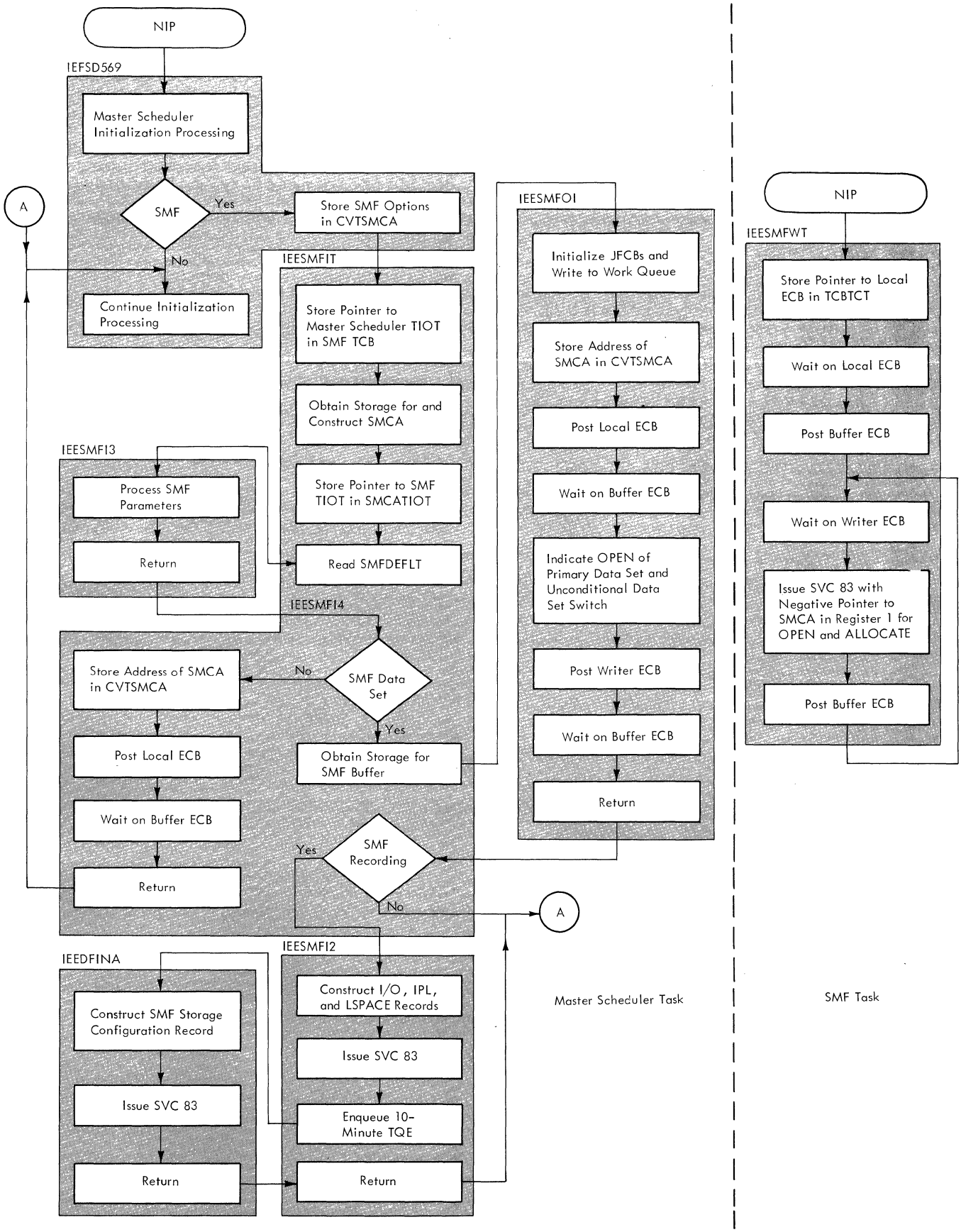


Figure 20. SMF Initialization Processing Flow

The other SMF initialization routines execute under the master scheduler TCB. SMF initialization routine IEESMFIT adds the SMF DD names to the master scheduler task input/output table (TIOT) and reads the SMF parameter member SMFDEFLT from SYS1.PARMLIB. SMF parameter processing routine IEESMFI3 processes the SMFDEFLT parameters and/or the SMF parameters entered from the console. SMF initialization routine IEESMFI2 initializes the 10-minute timer queue element (TQE) and constructs the SMF IPL and online device records. SMF open initialization routine IEESMFOI initializes the job file control blocks (JFCBs) for the SMF data sets and posts the SMF writer routine for the opening and allocation of these data sets. SMF storage configuration record creation routine IEEDFINA constructs the SMF storage configuration record (type 13).

When the SMF writer routine is dispatched, it stores the address of a local ECB in the TCBTCT field of the SMF TCB. (This local ECB provides communication between the master scheduler and SMF tasks. It is posted by IEESMFIT or by IEESMFOI after the address of the SMCA has been stored in the CVTSMCA field of the CVT.) The SMF writer routine then issues a WAIT macro instruction specifying the local ECB.

SMF initialization routine IEESMFIT receives control from master scheduler initialization routine IEFSD569 via a LINK macro instruction. When it is entered, it adds the compiled-in DD names SMFMANX and SMFMANY to the master scheduler TIOT and stores the address of the master scheduler TIOT in the SMCATIOT field of the SMCA. (For the format and description of the SMCA, see "Appendix A" in the MVT Job Management PLM.)

The SMF initialization routine then checks SYS1.PARMLIB for the existence of the SMF parameter member, SMFDEFLT. If it does not exist, the routine passes control to IEESMFI3 at entry point IEESMFMS. Otherwise, it opens the SYS1.PARMLIB data set and reads SMFDEFLT into main storage. If an I/O error occurs during the reading of the data set, IEESMFIT passes control to IEESMFI3 at entry point IEESMFI0.

When it has completed reading in the SMF parameter member, IEESMFIT passes control to IEESMFI3 to determine if all of the parameters have been entered correctly. If so, IEESMFI3 stores the SMF parameter values in the SMCA. If any required parameters are missing or incorrectly specified, or if IEESMFI3 was entered at IEESMFMS or IEESMFIO, the routine issues a WTOR macro instruction to request the operator to enter the SMF parameters to make the appropriate corrections. When he has entered the parameters correctly, IEESMFI3 stores the values in the SMCA. The routine then issues a WTO macro instruction to display the parameters. It then inspects the parameter associated with the OPI keyword. If the parameter is "YES", operator intervention is allowed. IEESMFI3 therefore issues a WTOR macro instruction to allow the operator to make changes. When the changes, if any, have been stored, IEESMFI3 returns control to IEESMFIT at entry point IEESMFI4.

IEESMFIT then determines if an SMF data set is specified by testing the SMCAMAN bit in the SMCAMISC field of the SMCA. If the bit is off, an SMF data set is not specified. In this case, IEESMFIT prepares to return to the master scheduler initialization routine, because SMF recording will not be possible. It stores the address of the SMCA in the CVTSMCA field of the CVT. It then issues a POST macro instruction specifying the local ECB to indicate to the SMF writer routine that the SMCA is initialized and its address stored in the CVT. IEESMFIT then issues a WAIT macro instruction specifying the buffer ECB (SMCABECB).

When the local ECB is posted the SMF writer routine has access to the SMCA. It, therefore issues a POST macro instruction specifying the buffer ECB, to indicate that the writer is ready and waiting for work. The SMF writer routine will now wait on the writer ECB (SMCAWECB).

When the buffer ECB is posted, the SMF initialization routine is activated. It returns control to master scheduler initialization routine IEFSD569.

If an SMF data set is specified, IEESMFIT continues with initialization processing. It issues a GETMAIN macro instruction specifying the system queue area to obtain main storage for the SMF buffer. It then passes control to SMF open initialization routine IEESFMOI via a LINK macro instruction.

The SMF open initializer prepares to have the SMF data sets opened and allocated. It compiles the JFCBs for the data sets (SYS1.MANX and SYS1.MANY) and uses the Queue Management Assign/Start and Read/Write routines to obtain space in the work queue data set and to write the

JFCBs.[1] It stores the JFCB addresses in the master scheduler TIOT and stores the address of the SMCA in the CVTSMCA field of the CVT. It indicates completion of this action to IEESMFWT by issuing a POST macro instruction specifying the local ECB. It then issues a WAIT macro instruction specifying the buffer ECB.

When the local ECB is posted the SMF writer routine has access to the SMCA. It therefore issues a POST macro instruction specifying the buffer ECB, to indicate that the writer is ready and waiting for work. The SMF writer routine stores pointers to the DCBs for the SMF data sets in the SMCA and then waits on the writer ECB.

When the buffer ECB is posted the SMF open initialization routine is activated. It sets the "first time switch" (SMCAFIRT) in the SMCA miscellaneous indicators field to specify that the primary data set is to be opened. It also sets the "unconditional switch bit" in the SMCASWA switches field to specify an unconditional data set switch. (This bit is set to indicate to the SMF writer routine that register 1 must be negative when the SVC 83 for opening and allocation of the SMF data sets is issued.) IEESMFOI then issues a POST macro instruction specifying the writer ECB to activate the SMF writer routine.

When the SMF writer routine determines that the "unconditional data set switch bit" is set, it loads register 1 with the address of the SMCA. It then sets register 1 negative and issues an SVC 83 for the opening and allocation of the SMF data sets. (The SVC processing for SMF initialization is described in the "Initialization and Restart" section of the MVT Job Management PLM.)

Upon return from the SMF SVC routines, the SMF writer routine issues a POST macro instruction specifying the buffer ECB, to indicate to IEESMFOI that opening and allocation are completed. It will now wait on the writer ECB until its services are again requested.

When the buffer ECB is posted, IEESMFOI is activated and it returns control to IEFSMFIT. The SMF initialization routine tests the SMCA miscellaneous indicators field to determine if SMF recording will be performed. If not, IEESMFIT returns control to master scheduler initialization routine IEFSD569.

If SMF recording is implemented, IEESMFIT passes control to SMF initialization routine IEESMFI2. This routine constructs the SMF IPL record (type 0) and the SMF online devices records (type 8). If volume accounting is specified, it also issues an SVC 78 for construction of the SMF LSPACE records (type 19). It issues an SVC 83 to have the records transferred to the SMF buffer.

IEESMFI2 then branches to the timer enqueue routine (IEAQTE00) in the timer second level interruption handler to add a compiled-in timer queue element (TQE), requesting 10-minute time intervals, to the timer queue. It then passes control to IEEDFINA via a LINK macro instruction.

IEEDFINA constructs the SMF storage configuration record (type 13). It issues an SVC 83 to have the record transferred to the SMF buffer. Upon return from the SVC routines, IEEDFINA returns to IEESMFI2, which returns to IEESMFIT, which returns to master scheduler initialization routine IEFSD569.

The SMF Writer Routine (IEESMFWT)

The SMF writer routine is the resident wait routine of the SMF task. It executes under a system TCB (the SMF TCB). It is initially involved in a WAIT/POST interchange with the SMF initialization routines executing under the master scheduler TCB. This is to ensure the SMF writer routine that the SMCA is initialized and its address stored in the CVT, before the writer routine performs its SMF processing.

When the SMCA is initialized, IEESMFWT issues a WAIT macro instruction specifying the writer ECB. The SMF writer routine regains control each time the ECB is posted, and performs one of the following functions:

- It requests opening of the SMF data sets.
- It requests data set switching.
- It writes the contents of the SMF buffer in the SMF data set.
- It determines if the SMF data set contains enough storage space for a record that must be written in segments.

These functions are implemented in the same manner for MFT and MVT. They are described in the "Common Elements of Job Management" section of the MVT Job Management PLM.

---

[1]The JFCB records appear as an incomplete queue entry. In the event of a system restart, the system restart routines return the space occupied by the JFCB records to the free-track queue, and the SMF open initialization routine replaces them when it is executed again.

Write-to-Programmer Facility

The user can invoke the write-to-programmer facility by specifying a routing code of 11 in the WTO/WTOR macro instruction (see the MFT Supervisor PLM for a discussion of the write-to-operator routine). When the write-to-operator routine (SVC 35) is entered, it examines all WTO/WTOR messages for a WTP routine code of 11. If such a code is specified, WTO passes control to the Write-to-programmer Initialization rouine (module IEFWTP00). This routine initializes registers and a work area before passing control to the Write-to-programmer Message Processing routine (module IEFWTP01). The WTP Message Processing routine uses the Transient Queue Management routines (SVC 90) to read, write, and assign SYS1.SYSJOBOF records for WTP messages. If an I/O error occurs in the job queue that is processing the WTP messages or if no record is available for a WTP message, the Write-to-programmer Error Processing routine (module IEFWTP02) receives control.

The three write-to-programmer routines are the same for the MFT and MVT configurations of the control program. For a detailed description of these routines, see the "Common Elements of Job Management" section of the MVT Job Management PLM.

APPENDIX A: TABLES AND WORK AREAS

This appendix contains descriptions and format diagrams of the major tables and work areas that are used by MFT job management. The tables and work areas are in alphabetical order, as shown below:

- Command Scheduling Control Block (CSCB)
- Data Set Enqueue (DSENQ) Table
- Interpreter Work Area (IWA)
- Job Control Table (JCT)
- Job File Control Block (JFCB)
- Job File Control Block Extension (JFCBX)
- Life-of-Task Block (LOT)
- Linkage Control Table (LCT)
- Master Scheduler Resident Data Area
- Partition Information Block (PIB)
- Small Partition Information List (SPIL)
- Step Control Table (SCT)
- Step Input/Output Table (SIOT)
- Task Input/Output Table (TIOT)
- Write-to-Programmer Control Block (WTPCB)

Tables and work areas are shown four or eight bytes wide for convenience, but are not necessarily drawn to scale. Tables that are stored in work queue entries are limited, by convention, to a length of 176 bytes.

The names of most fields are sufficient to describe the fields; those that require further explanation are described in the text accompanying the table. Where a macro instruction may be used to include a DSECT of a table in routines using the table, the name of the mapping macro instruction is also given. The displacement of each field is shown to the left of each table; the values in parentheses show the hexadecimal displacement.

COMMAND SCHEDULING CONTROL BLOCK (CSCB)

Description: A command scheduling control block (CSCB) (Figure 21) is an area for communications between the command scheduling routine (SVC 34) and the command execution routines. Input CSCBs are created by several system routines. When an input CSCB is created, it is placed in a chain of CSCBs by the command scheduling routine. It remains in the chain until it is deleted from the chain by the command scheduling routine, which may also free the main storage occupied by the CSCB. An input CSCB is created under the following circumstances:

- A CSCB is created by the command scheduling routine each time a task-creating command is encountered. If the task is a reading or writing task, the CSCB is deleted from the chain, and its main storage released, when the task terminates.
- A CSCB is created by the queue management dequeue routine each time the initiator dequeues a job. This CSCB is deleted from the chain, and its main storage released, when the last step of the job has terminated.
- A CSCB is created by a system output writer each time it encounters a DSB that was not preceded by another DSB in the current queue entry. The CSCB serves as a communication area, allowing the cancelation (by operator command) of the subtasks established by the writer. The CSCB is deleted from the chain, and its main storage released, when the writer encounters an SMB (or the last block in the current queue entry).

A control CSCB is updated (and changed to the control format if necessary) by the command scheduling routine when a CANCEL jobname (job selected), CANCEL writer device, MODIFY, or STOP command is encountered.

Although most of the fields are self-explanatory, the following require further description:

- Verb Code: This byte contains a code corresponding to the command for which the CSCB has been created.

- Status Flags: This byte indicates the status (pending/not pending) of the CSCB, and the action to be taken by the command scheduling routine. In addition to command processing, the command scheduling routine may be entered to add the CSCB to the chain, delete it, free its main storage, or to branch to the abnormal termination routine.

Name	Bit	Setting	Meaning
CHAP	0	1	Assignment pending
CHSYS	1	0	Problem program CSCB
		1	System task CSCB
CHSOUT	2	1	Cancel all SYSOUT
CHQSPC	3	1	Insufficient queue space for 422 ABEND
CHAD	4	1	Add this CSCB to the chain
CHDL	5	1	Delete this CSCB from the chain
CHFC	6	1	Free this CSCB's core
CHABTERM	7	1	Execute branch entry to ABTERM

- Activity Flags: This byte indicates the type of activity with which the CSCB is associated.

Name	Bit	Setting	Meaning
CHSWAP	0	1	Job qualifies to be swapped (TSO only)
CHTERM	1	1	Terminal job
CHRESA	2		Reserved
CHRESB	3		Reserved
CHCL	4	1	Cancelable job step
CHCLD	5	1	Cancel communication switch
CHAIFX	6	1	Cancelable (MFT only)
CHIFY	7	1	System assigned procedure (MFT only)

- Communication Flags: This byte indicates the function to be performed by the command processing routine.

Name	Bit	Setting	Meaning
CHRESC	0		Reserved
CHJCT	1	1	Reader return with in-core JCT
CHPSD	2	1	Writer pause data set
CHPSF	3	1	Writer pause forms
CHAC	4	1	ID specified on START command
CHDSI	5	0	Data set integrity
		1	No data set integrity
CHHIAR	6	0	H0 specified on START command
		1	H1 specified on START command
CHDEF	7	0	Use hierarchy specified by CHHIAR
		1	Default to H0

- Express CANCEL Flags: This byte indicates the parameters passed with the CANCEL command to the CANCEL processor.

Name	Bit	Setting	Meaning
CHALL	0	1	ALL specified
CHINN	1	1	IN specified
CHOUT	2	1	OUT specified
CHHOLD	3	1	HOLD queue specified
CHQUE	4	1	Specific queue specified
CHDUMP	5	1	Dump specified
CHJB	6	1	End scan
CHRESD	7		Reserved

- MFT STC Switches: This byte contains codes set by the reader/interpreter control routine of System Task Control.

Code	Meaning
X'00'	No errors for started system task
X'80'	JCL error for started system task
X'81'	I/O error for started system task
X'C0'	No errors for started problem program
X'C4'	JCL error for started problem program
X'C8'	I/O error for started problem program

Mapping Macro Instruction: IEECHAIN

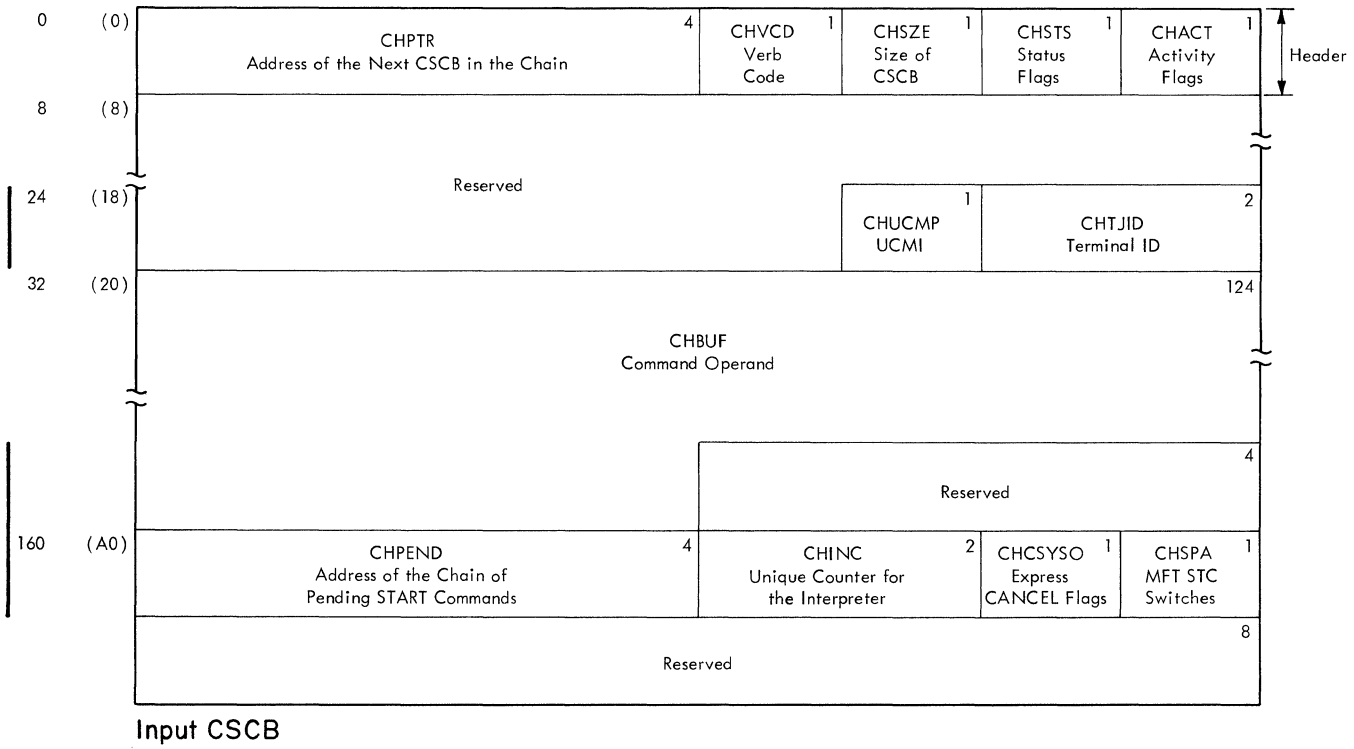


Figure 21. Command Scheduling Control Block (CSCB) (Part 1 of 2)

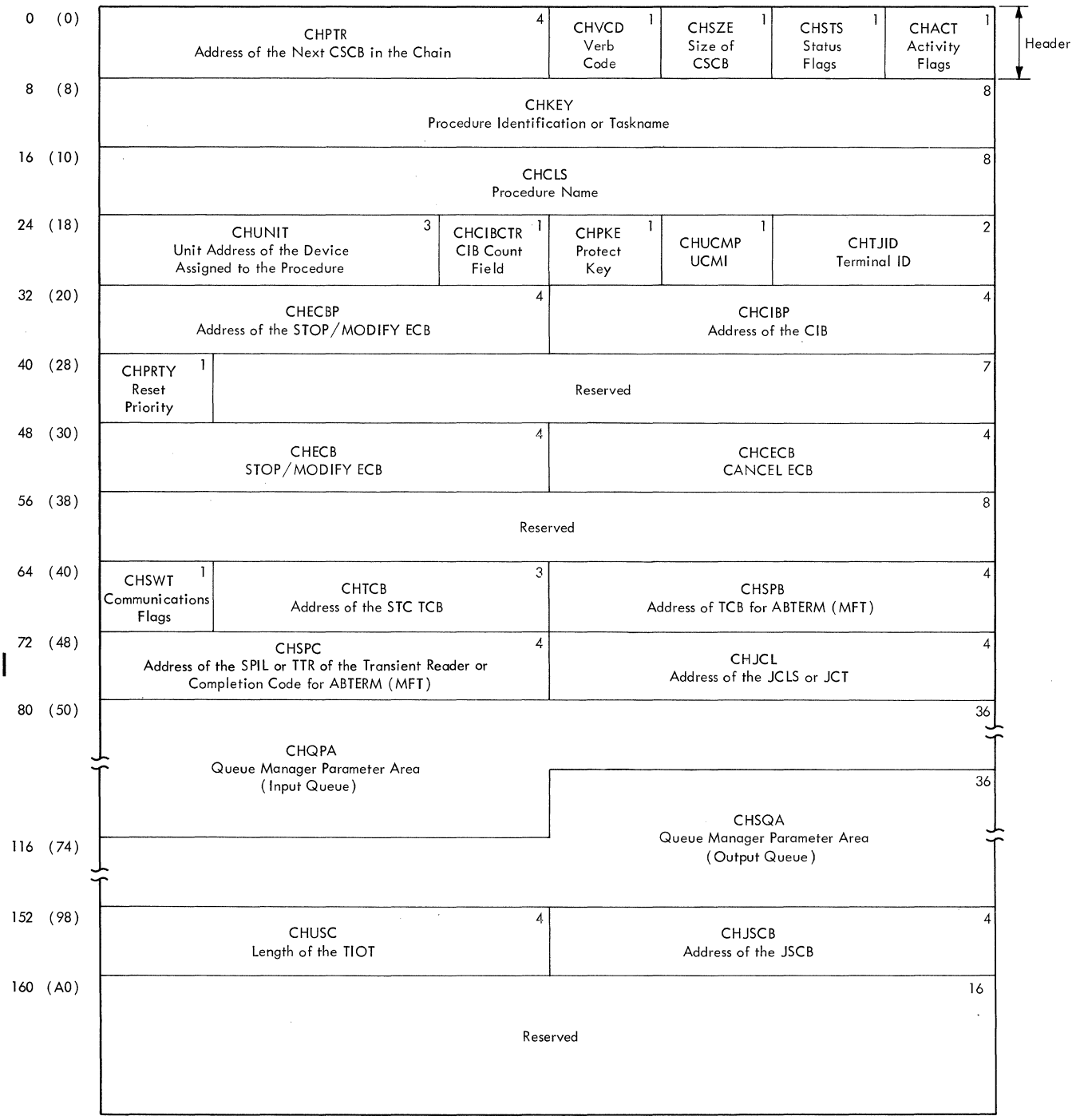
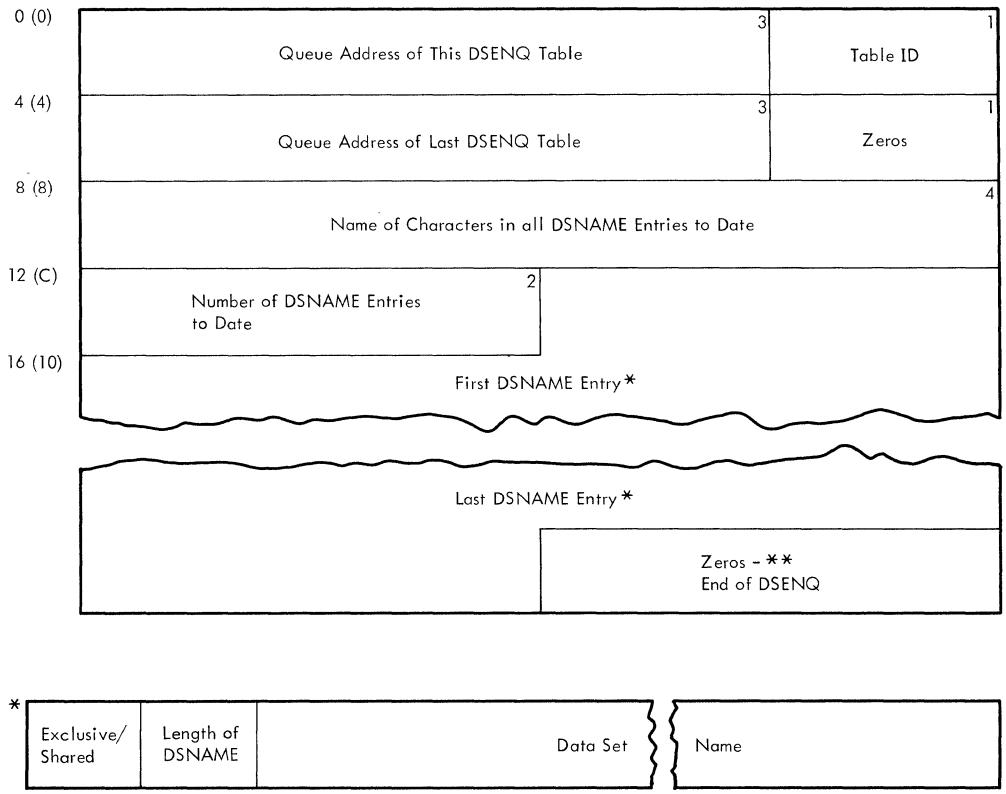


Figure 21. Command Scheduling Control Block (CSCB) (Part 2 of 2)

DATA SET ENQUEUE TABLE (DSENQ)

Description: The data set enqueue table (DSENQ) (Figure 22) is built by the DD statement processor routine of the interpreter, and is used by the initiator to construct an ENQ macro instruction parameter list to prevent routines performing different tasks from using the same exclusive data sets concurrently. The table contains an entry for each data set (except temporary data sets) required for a job.



** If the last entry uses the last available space in the tables but no overflow occurs, the zero bytes are omitted.

Figure 22. Data Set Enqueue Table (DSENQ)

INTERPRETER WORK AREA (IWA)

Description: The 2048-byte interpreter work area (IWA) (Figure 23) is obtained from subpool zero by a GETMAIN macro instruction in the interpreter initialization module (IEFVH1). The IWA contains information used by the interpreter routines; it is the area in which job description tables are built before they are placed in the work queues.

Although most of the fields in the interpreter work area are self-explanatory, the following require further description:

- Default Parameters: The PARM field of the EXEC statement in the reader procedure contains parameters to be used when no explicit specification is made. These parameters specify whether the installation requires a programmer's name or account number on each JOB statement, the priority to be assigned to a job if no priority has been specified, whether commands in the input stream should be processed (or ignored), and the device, primary quantity, and secondary quantity to be allocated to system output data sets.

- Switches A-I: These fields contain internal switches used for communicating status information among the interpreter routines.

Switch A:

Name	Bit	Setting	Meaning
JTOP	0	1	Job to process
JHS	1	1	Job has a step
JCTTQ	2	1	JCT to put on queue
SCTTQ	3	1	SCT to put on queue
DFSH	4	1	Data flush
JFSH	5	1	Job flush
EOFR	6	1	End-of-file received
SAFSH	7	1	Flush to a /*

Switch B:

Name	Bit	Setting	Meaning
CXP	0	1	Continuation expected by Scan
CXPN	1	1	Continuation expected and not received
CXPC	2	1	Continuation expected and canceled
CANDD	3	1	DD * generated
DDAST	4	1	DD * or DD data
DDATA	5	1	DD data
FRCV	6	1	First statement received
SFJN	7	1	Search for job name

Switch C:

Name	Bit	Setting	Meaning
JCTRTN	0	1	CSCB return
IOERR	1	1	I/O error on input
NRCV	2	1	Null statement received
PEXP	3	1	Procedure EXEC statement expected
VOLTQ	4	1	Volume table to put on queue
DSNTQ	5	1	Data set name table to put on queue
PLSMB	6	1	Put last SMB for this step on queue
QMERR	7	1	Queue manager I/O error

Switch D:

Name	Bit	Setting	Meaning
JOBROLLF	0	1	Roll on job statement
JOBREGNS	1	1	Region on job statement
FEXRCV	2	1	First EXEC received this job
FDDRCV	3	1	First DD received this job
DBFST	4	1	First entry to DSENQ
DBLST	5	1	Last entry to DSENQ
DCTFST	6	1	First dictionary entry received
SYMPRC	7	1	First access of a procedure

Switch E:

Name	Bit	Setting	Meaning
PROC	0	1	Procedure library being used
GPI	1	1	Get procedure library input
PREF	2	1	Procedure library end-of-file
PRCV	3	1	Prime procedure buffer
CONCAT	4	1	Concatenation in merge
POVRD	5	1	Override procedure DD statement
POVRX	6	1	Override procedure EXEC statement
	7		Unused

Switch F:

Name	Bit	Setting	Meaning
ORPARMOR	0	1	Parameter override
ORPARMBL	1	1	PARM parameter present
ORCONDOR	2	1	Condition override
ORTIMEOR	3	1	TIME override
ORTIMEO	4	1	TIME = zero
ORACTOR	5	1	ACCT override
ORREGOR	6	1	Region override
ORROLLOR	7	1	Roll override

Switch G:

Name	Bit	Setting	Meaning
ORRDOR	0	1	Reader override
ORSDPOR	1	1	Step dispatching priority override
	2-7		Unused

Switch H:

Name	Bit	Setting	Meaning
PCPCOMM	0	1	PCP working on command
RDRDCBO	1	1	Reader opened
PROCDCBO	2	1	Procedure library opened
CPSYSFLG	3	1	Checkpoint restart EXEC statement
CPFLGXX	4	1	Reserved for checkpoint restart
PROCSW	5	1	Statement invokes a procedure
CPSTPFL	6	1	Checkpoint restart step flush
PCPSYSIN	7	1	SYSIN DD * encountered in PCP

Switch I:

Name	Bit	Setting	Meaning
BLKPRC	0	1	Block procedure library
IWABAS	1	1	Bypass Assign/Start
IWADDNM	2	1	DDNAME = Key this card
IWAKGSW	3	1	Blocked procedure PCP
BLKMLTER	4	1	Procedure library blocksize
DSNLIT	5	1	DSN = 'LITERAL'
	6		Reserved
SPOOLDD	7	1	DD * or data indicator

- Switch K: This field contains the Priority Change Value for the CHAP macro instruction.
- Switch L: This field contains the Default Allocation level in MSGLEVEL.
- Switch M: This field contains the Default JCL level in MSGLEVEL.
- Switch N: This field contains the length of the fixed part of the message for symbolic parameter substitution.
- Switch X1: This field is set to X'80' for a search of the DDNAME reference table or to X'40' for SYSOUT.

- Checkpoint/Restart Switches: These fields contain switches that communicate checkpoint/restart status information to the interpreter routines.

CHECKPOINT RESTART SWITCHES A:

Name	Bit	Setting	Meaning
	0-1		Unused
JOBRDNR	2	1	RD=NR
JOBRDNC	3	1	RD=NC or RD=RNC
JOBRDR	4	1	RD=R or RD=RNC
	5-7		Unused

CHECKPOINT RESTART SWITCHES B:

Name	Bit	Setting	Meaning
CPFLG	0	1	GET/FREE SYSCHECK DD statement core
	1		Unused
CPDUM	2	1	Dummy step control table required step flush
CRRES1	3		Reserved
CRRES2	4		Reserved
CRRES3	5		Reserved
CRRES4	6		Reserved
CRIMRS	7	1	Immediate restart (PCP)

- Scan Switches: This field contains internal switches used by the Scan routines.

Name	Bit	Setting	Meaning
RPRSW	0	1	Right parenthesis switch
PDELSW	1	1	Period delimiter switch
ASTSW	2	1	Asterisk switch
FLUSHSW	3	1	Flush switch
LDL	4	1	Last delimiter switch
DCBSW	5	1	DCB switch
JGC	6	1	Text sublist switch
FERROR	7	1	Error switch

- Control and Scan Joint Switches: This field contains switches set by the Control routines to pass information to the Scan routine.

Name	Bit	Setting	Meaning
CMT	0	1	Comment switch
DDOV	1	1	DD override switch
ENDS	2	1	End scan switch
COLST	3	1	Column 72 (continuation) switch
JOBSW	4	1	JOB switch
EXECSW	5	1	EXEC switch
DDSW	6	1	DD switch
SNPSW	7	1	Statement SYSOUT switch

- Exit Switches: This field contains switches indicating conditions which cause exits to user routines.

Name	Bit	Setting	Meaning
	0-3		Unused
IWATRKS	4	1	Track stacking
IWAQFIOE	5	1	Job queue full
IWASFIND	6	1	Special procedure library FIND
IWAQENTR	7	1	Special queue manager entry

- System Input Allocation Table: This area contains a list of pointers to the UCBs corresponding to units available for allocation to system input data sets.

- Data Delimiter: This area contains the valid DLM value.

- Queue Address Table: This area contains the addresses (in TTR form) of the next two records assigned to the job's input queue entry, and the addresses (in TTR form) of the first joblib SIOT, the first scan dictionary record, and the DD override table.

- Input Stream Parameter List: This area describes the statement last encountered in the input stream, and contains a pointer to the field currently being processed.

- Procedure Library Parameter List: This area describes the statement last read from the procedure library, and contains a pointer to the field currently being processed.

- Procedure Library Merge Control Data: This area contains information used in merging statements from the input stream with statements from the procedure library. The information includes the statement names, the step names, and the names of the previous and next procedure steps.

Mapping Macro Instruction: IEFVMIWA

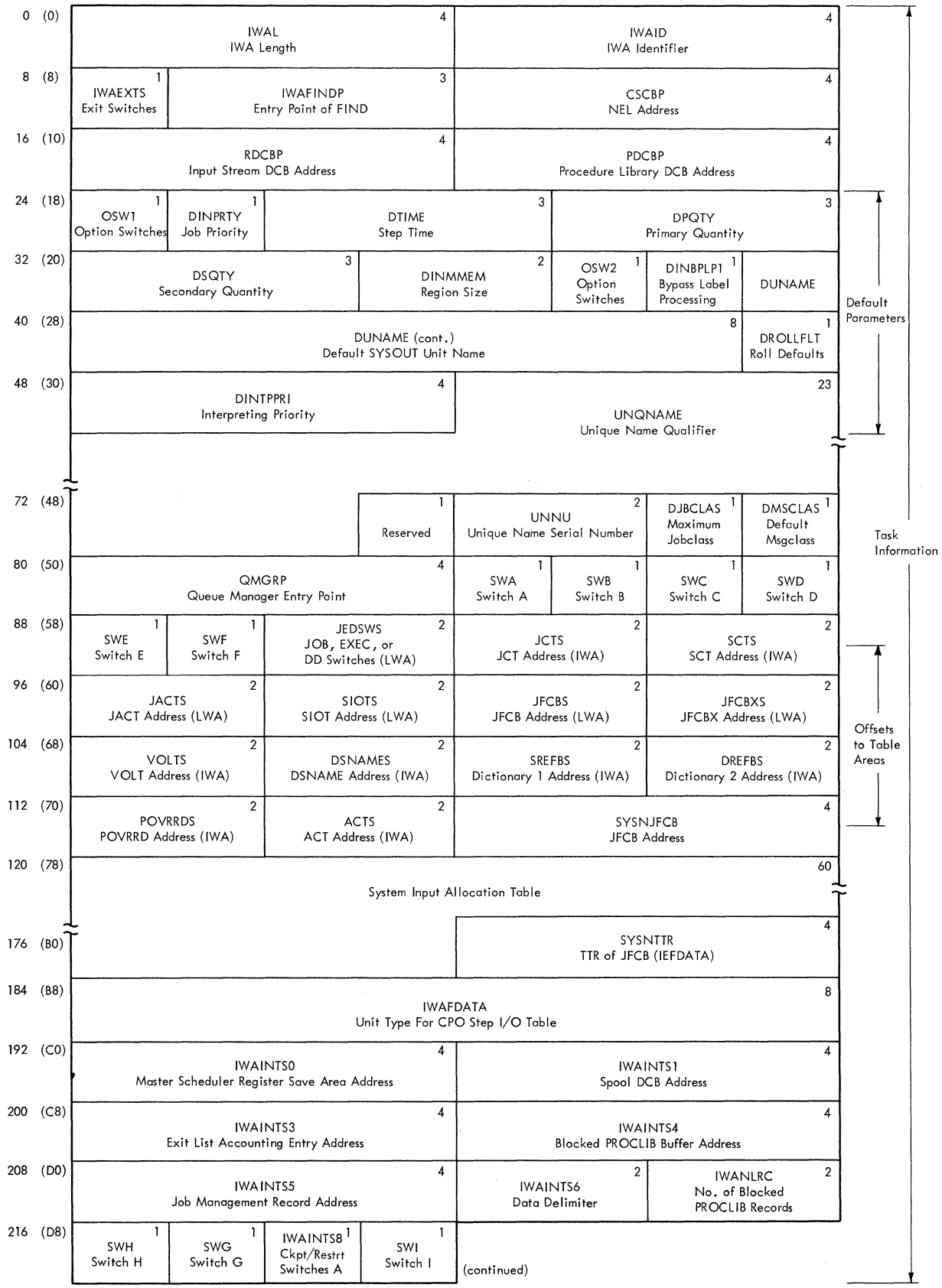


Figure 23. Interpreter Work Area (IWA) (Part 1 of 4)

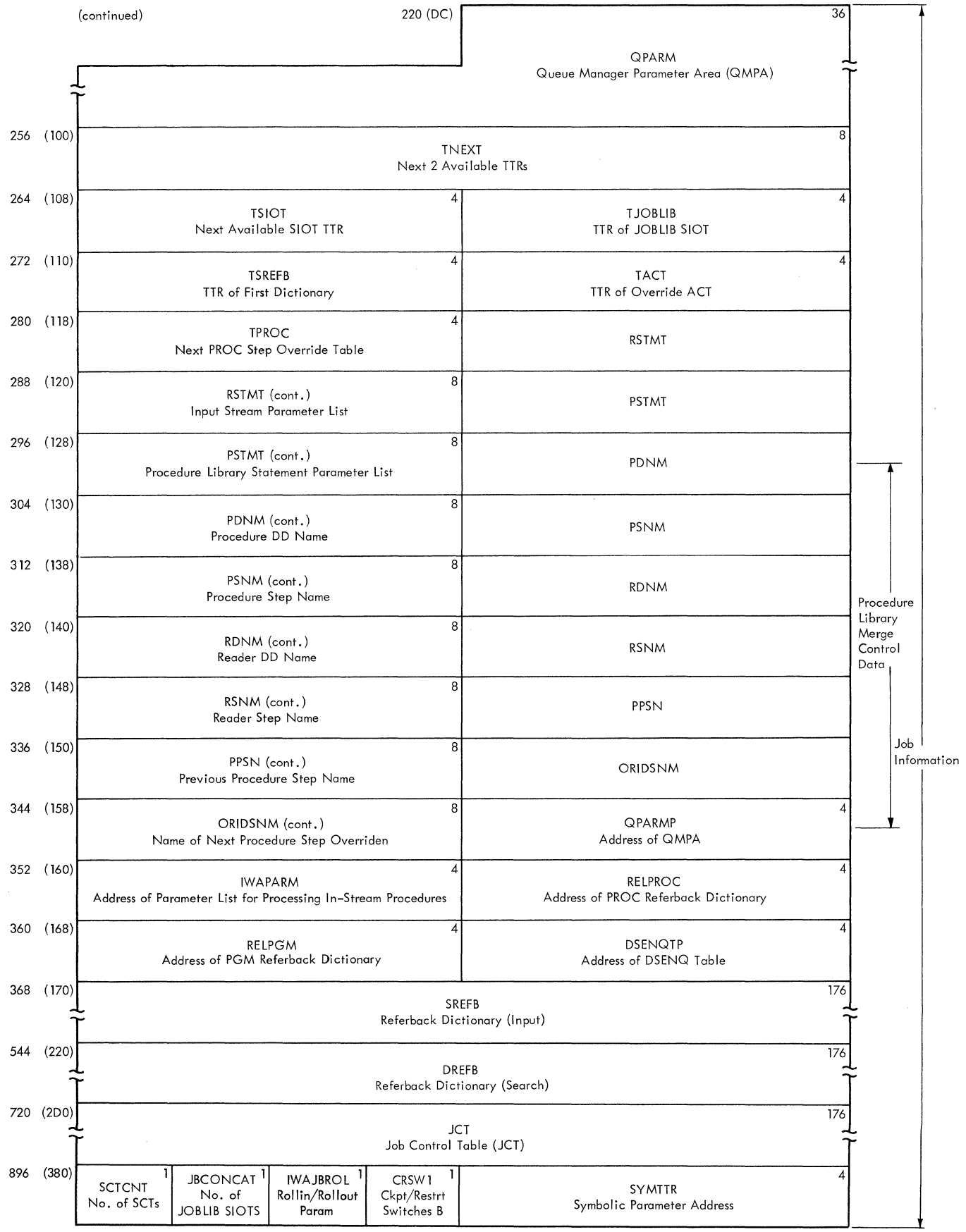


Figure 23. Interpreter Work Area (IWA) (Part 2 of 4)

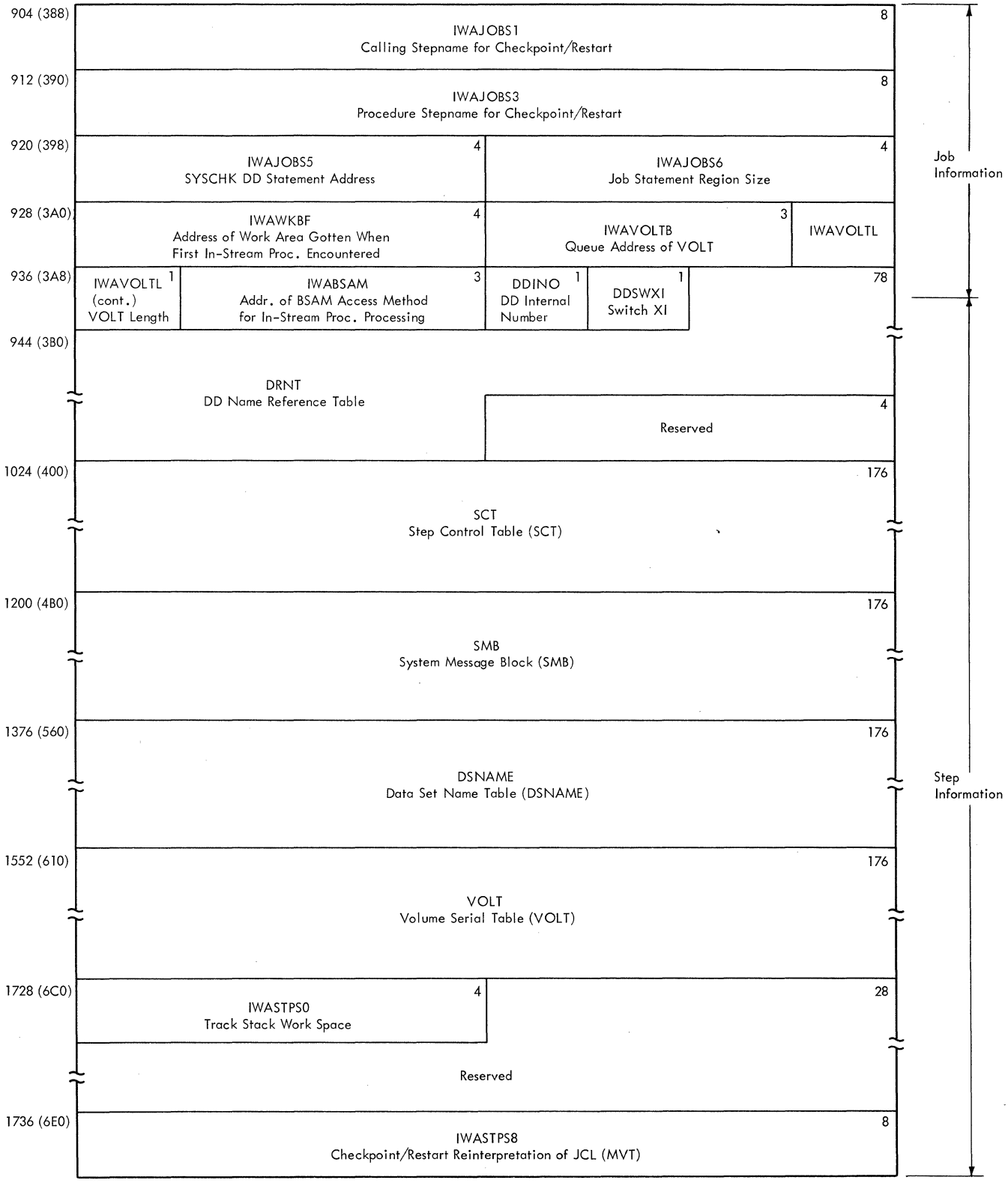


Figure 23. Interpreter Work Area (IWA) (Part 3 of 4)

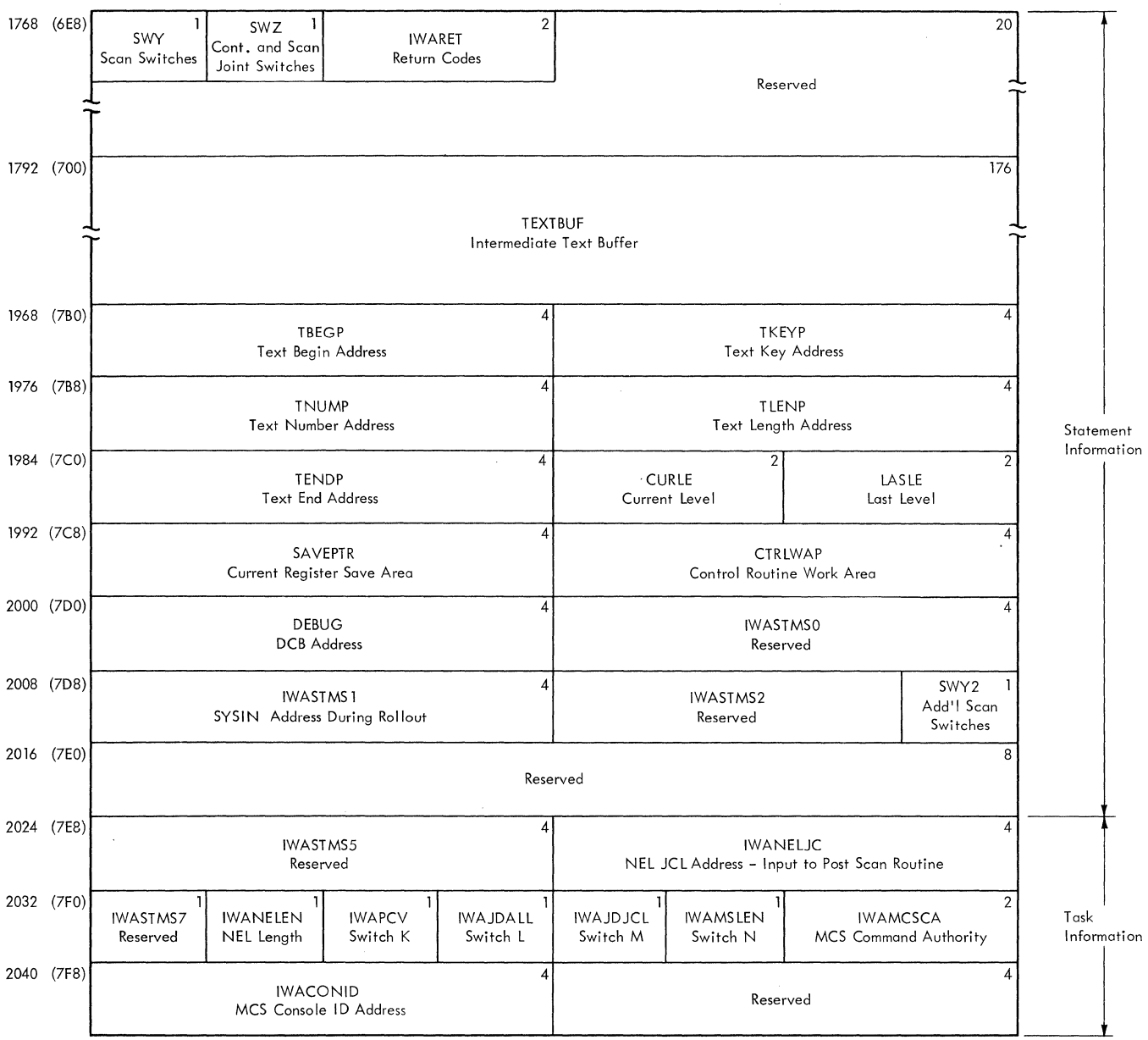


Figure 23. Interpreter Work Area (IWA) (Part 4 of 4)

JOB CONTROL TABLE (JCT)

Description: The job control table (JCT) (Figure 24) is created in the interpreter work area by the job statement processor routine of the interpreter. It contains information from the JOB statement, job status information, and pointers to other tables in the job's input queue entry. When the interpreter has processed all steps of a job, the JCT is written into the appropriate input queue according to priority; it is read back into main storage by the initiator job selection and job delete routines.

Although most of the fields in the job control table are self-explanatory, the following require further description:

- Job Status Indicators: The sixth byte of the JCT indicates the status of the job as shown below:

Bit	Setting	Meaning
0	1	A JOBLIB DD statement is included with the job
1	1	Job flush
2	1	Job step canceled by condition codes
3	1	Step flush
4	1	JCT ABEND
5	1	Job failed
6	1	Job includes a cataloged procedure
7	1	Job is a "no setup" job

- Additional Job Status Indicators: The byte indicates the status of the job as follows: Bit 0 is set to 1 to indicate spooled SYSIN data for the job. Bits 1 through 7 are reserved.

- Checkpoint/Restart Indicators: This two byte field indicates the checkpoint/restart status as shown below:

Byte 1

Bit	Setting	Meaning
0	1	Warm start
1		Not used by MFT
2		Not used
3	1	Checkpoint taken for this step
4	1	Intra-step checkpoint/restart to be done
5	1	Step restart to be done
6-7	0	Must be set to zero

Byte 2

Bit	Setting	Meaning
0	1	SYSCHK DD statement is included with the job
1	1	RD keyword parameter is not NC
2	1	No restart is to be done
3	1	No checkpoints are to be taken
4	1	Do restart if necessary
5	1	Direct SYSOUT writer active at checkpoint
6	1	Job is eligible for direct SYSOUT facilities.
7	1	DSDR processing has not successfully ended

- SYSOUT Classes: The first 36 bits of the five-byte field are used to indicate the system output classes that contain data. The four remaining bits are reserved.

Mapping Macro Instruction: IEFAJCTB

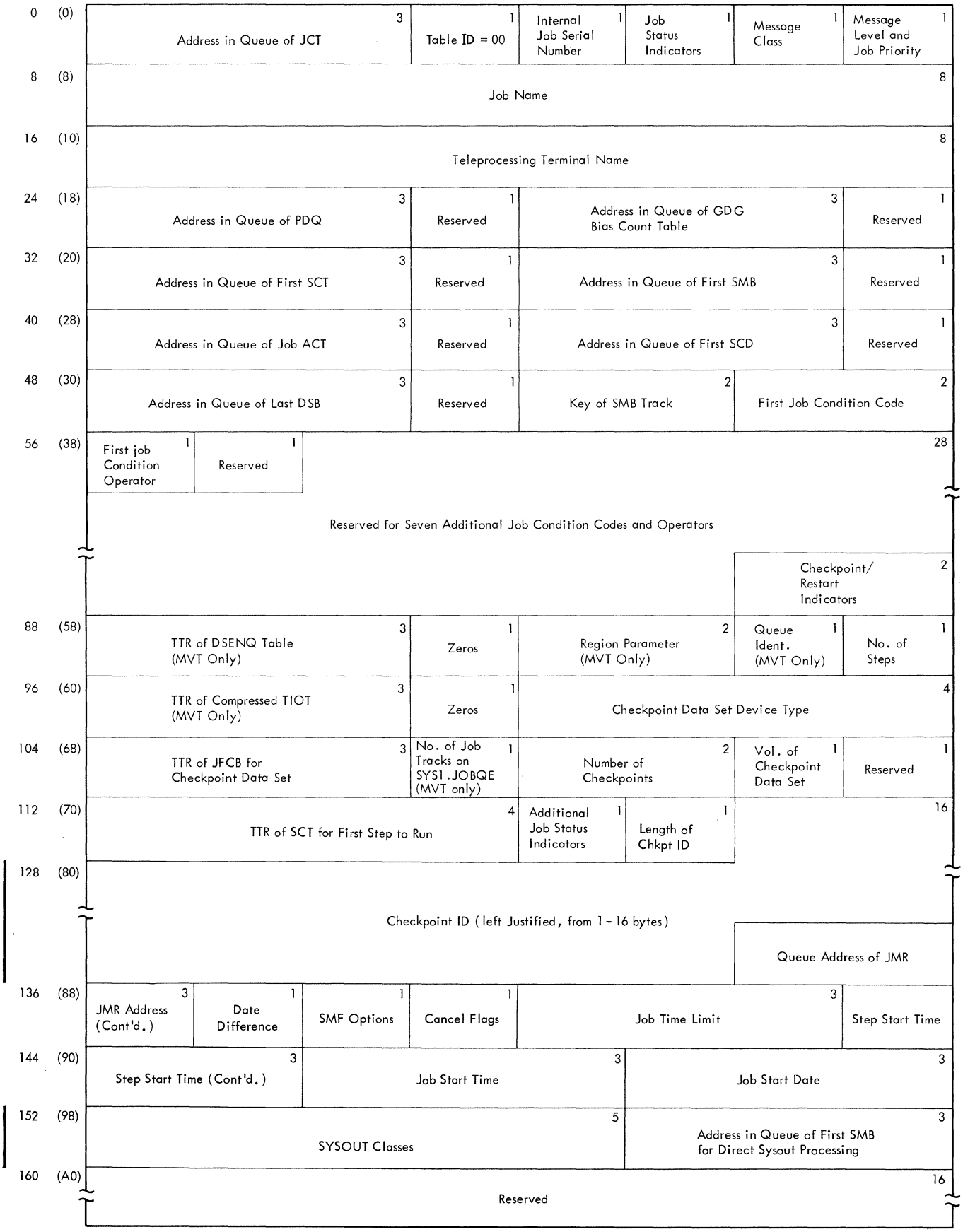


Figure 24. Job Control Table (JCT)

JOB FILE CONTROL BLOCK (JFCB) AND EXTENSION (JFCBX)

Description: A job file control block (JFCB) (Figure 25) is constructed in subpool zero (from information in a DD statement) by the interpreter DD statement processor routine. The JFCB is written into the job's input queue entry, and retrieved when a DCB with the corresponding name is opened. The information in the JFCB, which describes the characteristics of a data set, may be modified by the open routine.

A JFCB contains enough space to record five volume serials. If more than five volume serials are specified, enough job file control block extensions (JFCBXs) to contain the additional volume serials are constructed; each JFCBX can contain up to fifteen additional volume serials.

Additional information on the contents of the JFCB and JFCBX may be found in the publication, IBM System/360 Operating System: System Control Blocks, GC28-6628.

Mapping Macro Instruction: IEFJFCBN

LIFE-OF-TASK (LOT) BLOCK

Description: The 384-byte life-of-task (LOT) block (Figure 26) is built in a main storage area obtained from subpool 253. It contains information for scheduling functions, and is used by system task control and initiators. It is created by the Job Select module for initiating problem programs.

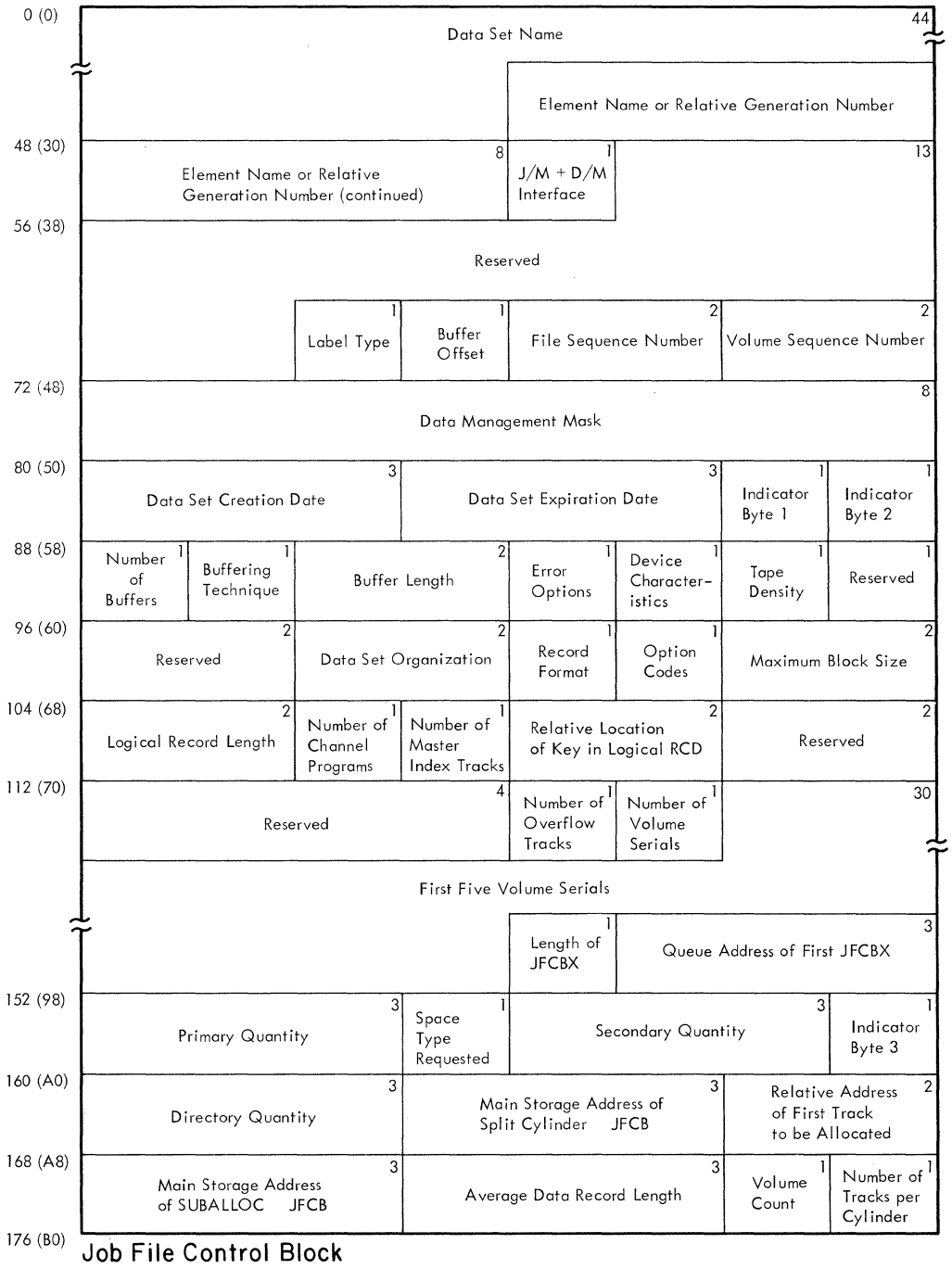
The LOT block contains the linkage control table (LCT), a two-level register save area (REGSAVE), an input queue manager parameter area (QMGR1), an output queue manager parameter area (QMGR2), the address of the ECB list, the address of the PIB, the address of the SPIL, and the ECB List.

LINKAGE CONTROL TABLE (LCT)

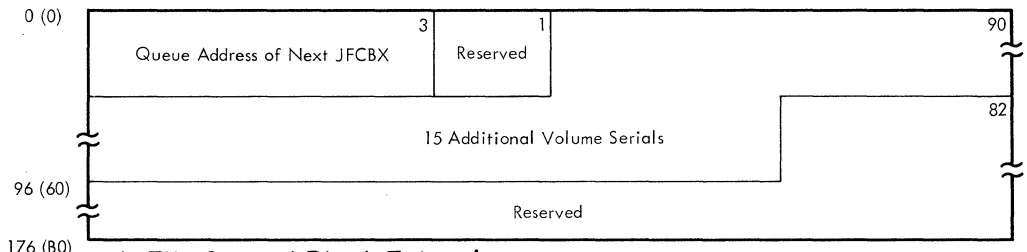
Description: The linkage control table (LCT) (Figure 27) is part of the LOT block constructed by the Job Select module in subpool 253. It is also built separately by System Task Control, in which case its storage is obtained from subpool zero. It is a communications area used by the routines of the Initiator, System Task Control, Allocation, and Termination.

Most of the fields in the LCT are self-explanatory; it should be noted, however, that the job termination status bit is the low-order bit of the one-byte device features field.

Mapping Macro Instruction: IEFALLCT



Job File Control Block



Job File Control Block Extension

Figure 25. Job File Control Block (JFCB) and Extension (JFCBX)

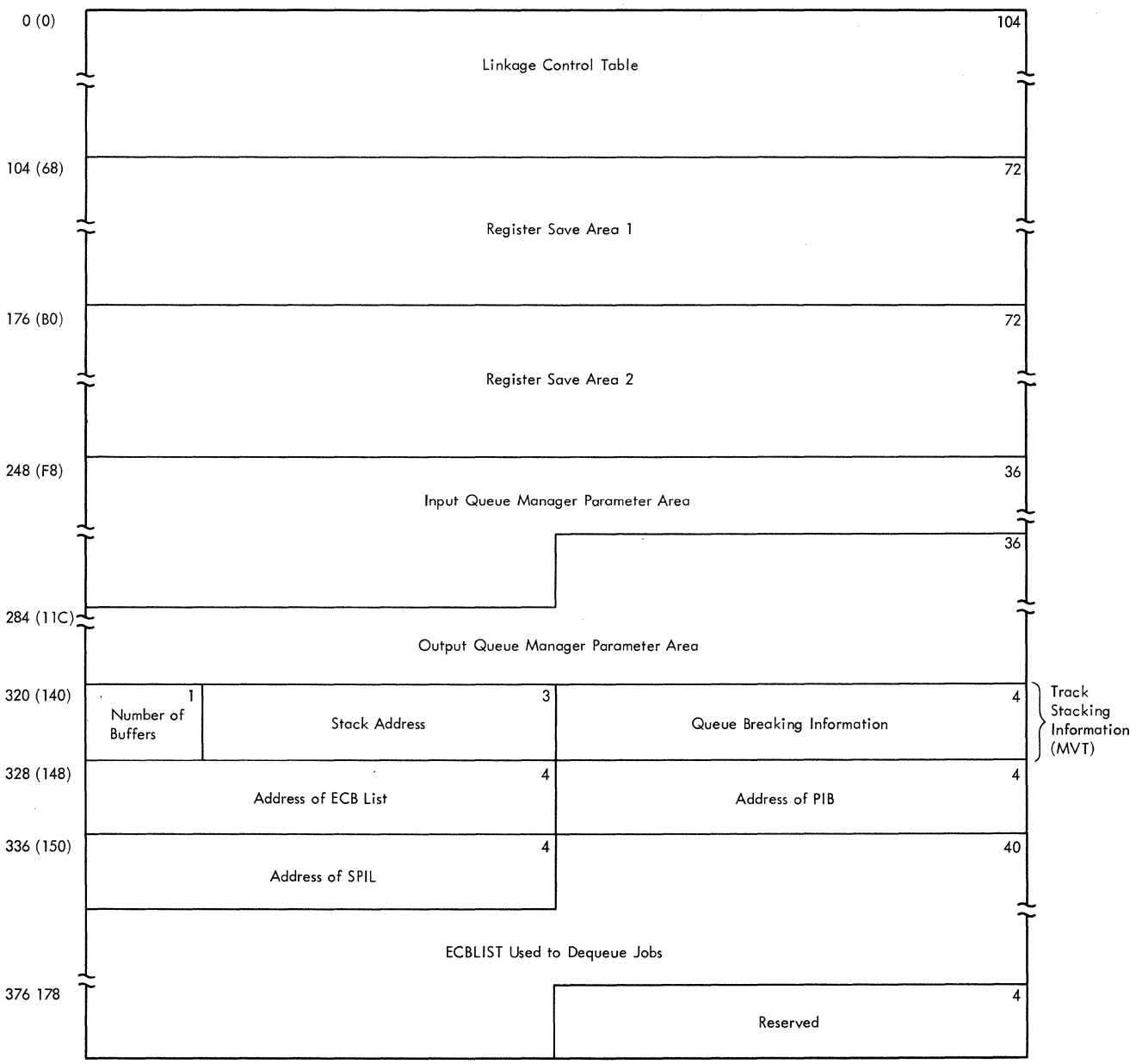


Figure 26. Life-of-Task (LOT) Block

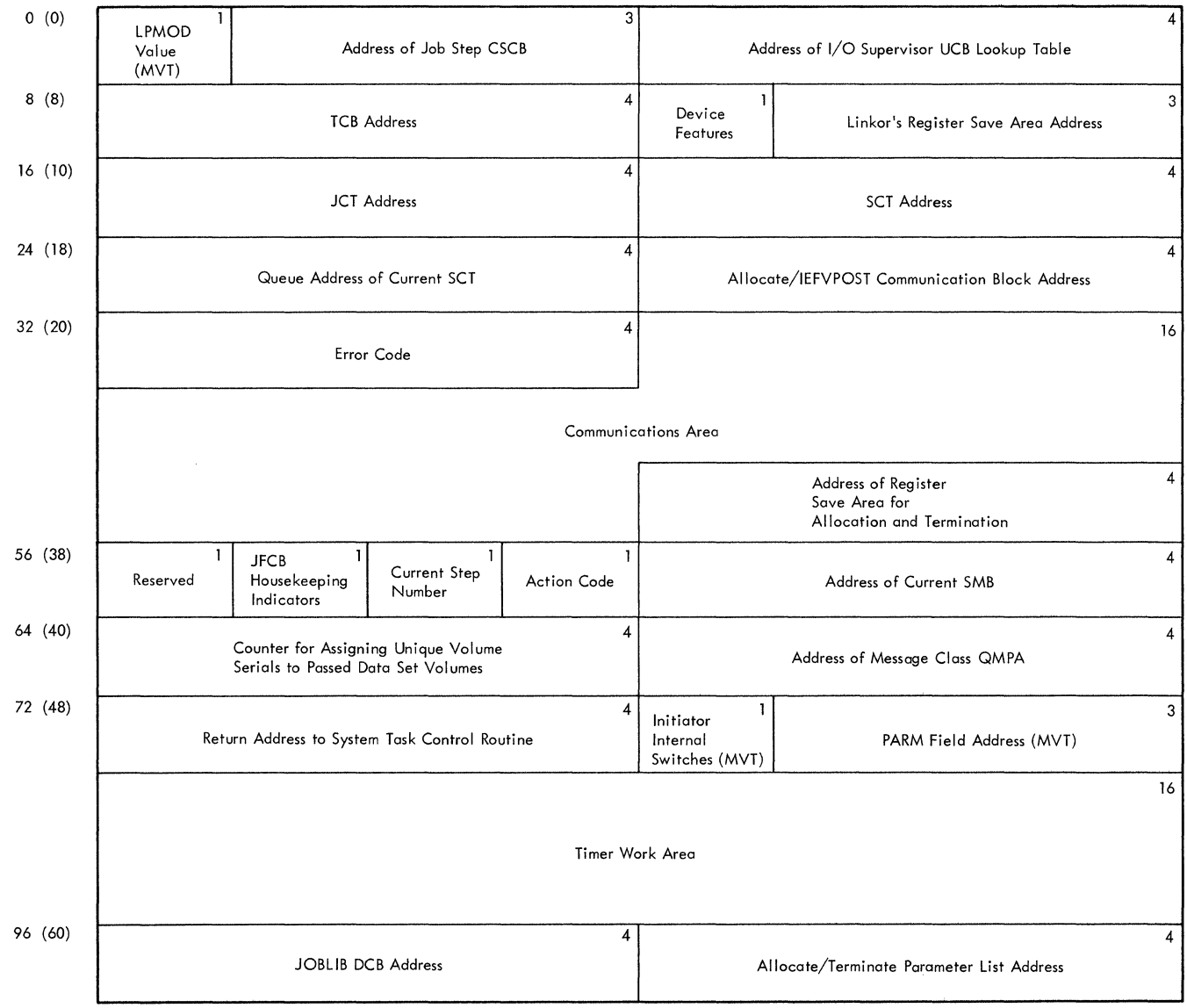


Figure 27. Linkage Control Table (LCT)

MASTER SCHEDULER RESIDENT DATA AREA

Description: The master scheduler resident data area (Figure 28), which is in the nucleus area of main storage, contains information used by the queue initialization, command scheduling, initiator, and I/O device allocation routines. Its location is stored in the CVTMSER field of the communication vector table.

Most of the fields in the master scheduler resident data area are self-explanatory; those fields that require further explanation are described below:

- Queue Formatting Switch: If the high-order bit of this field is on, it indicates that the queue data set must be formatted.

- Transient Reader TTR: This field is used by the transient reader suspend routine to store the address of the work queue data set where the reader information was placed when the reader was suspended.

- DEFINE Control Information: If the high-order bit of this field is on, it is a DEFINE operation; if off, it is IPL time. The second bit indicates that a list of the partitions' sizes and job class(es) has been requested; the third bit indicates that there is an adjacent partition check; the fourth bit is set when initialization is complete to allow DEFINE commands to be accepted; the fifth bit is set on when the operator has requested partition changes at IPL; the sixth bit indicates that a small partition cannot terminate because of the DEFINE operation; the seventh bit indicates that a DEFINE command has been issued during operation; the eighth bit indicates that the system has storage protection.

- Status Flags: When set on, status flags indicate:

Bit	Meaning
0	System Initialization in progress
1	MONITOR JOBNAMES
2	Reserved
3	VARY/UNLOAD summary
4	Queue hold-release
5	DISPLAY ACTIVE processing
6-7	Reserved

- Log Status Flags:

Bit	Meaning
0	Log Data Set Sysout Scheduling
1	Log Threshold Reached

- MFT Switches: When set on, flags indicate:

Bit	Meaning
0	Transient Reader Active
1	Transient Reader in Core
2	Pending START command for transient reader
3	MFT Environment switch
4	System Assigned Reader is Running
5	Core storage is in System

- Initialization Switches: When set on, flags indicate:

Bit	Meaning
0	IPL switch
1	SYSOUT IPL
2	SYSOUT job start
3-4	Reserved
5	34 Security
6	Queue initialized
7	Procedure catalog initialized

- System Exclusive Switches: When set on, switches indicate:

Bit	Meaning
0	Console flag (PCP only)
1	CANCEL flag for ABEND (PCP only)
2	Roll-out flag (PCP only)
3	Spinoff flag (PCP only)
4	MONITOR data set name
5	MONITOR space

- Pending Flags: When set on, flags indicate:

Bit	Meaning
0	IPL Date
1	Region busy
2	Command move completed
3	Interpreter command return
4	System Input control purge request
5	System output control purge request
6	Blank start pending (REQ=1,START BLANK=0)
7	Console command suppressed by WTO/WTOR Exit Routine

- ECB Flags: When set on, flags indicate:

Bit	Meaning
0	External interrupt
1	WTO or WTOR
2	WTL
3	Console Attention key hit
4	System Input
5	System Output
6	Master command routine
7	Summary bit, Vary UCB scan required

- Resident Switches: When set on, switches indicated:

Bit	Meaning
0	IPL has been completed
1	WTO or WTOR pending
2	Console usage, Primary or alternate
3	Log purge request
4	Reader has reached end of file, or Start reader
5	New reader pending
6	New writer pending New writer pending (Modify)
7	Job notification (1=yes)

- Fetch Flags: When set on, flags indicate:

Bit	Meaning
0	Named Fetch
1	Defer current command execution sequence
2	TCB Tree Trace Fetch (Locate)
3	Auxiliary FETCH given
4	Reply bit to Request attention
5	Pseudo-SYSOUT flag
6	MONITOR STATUS
7	Queue hold-release

- Mapping Macro Instruction: IEEBASEB.

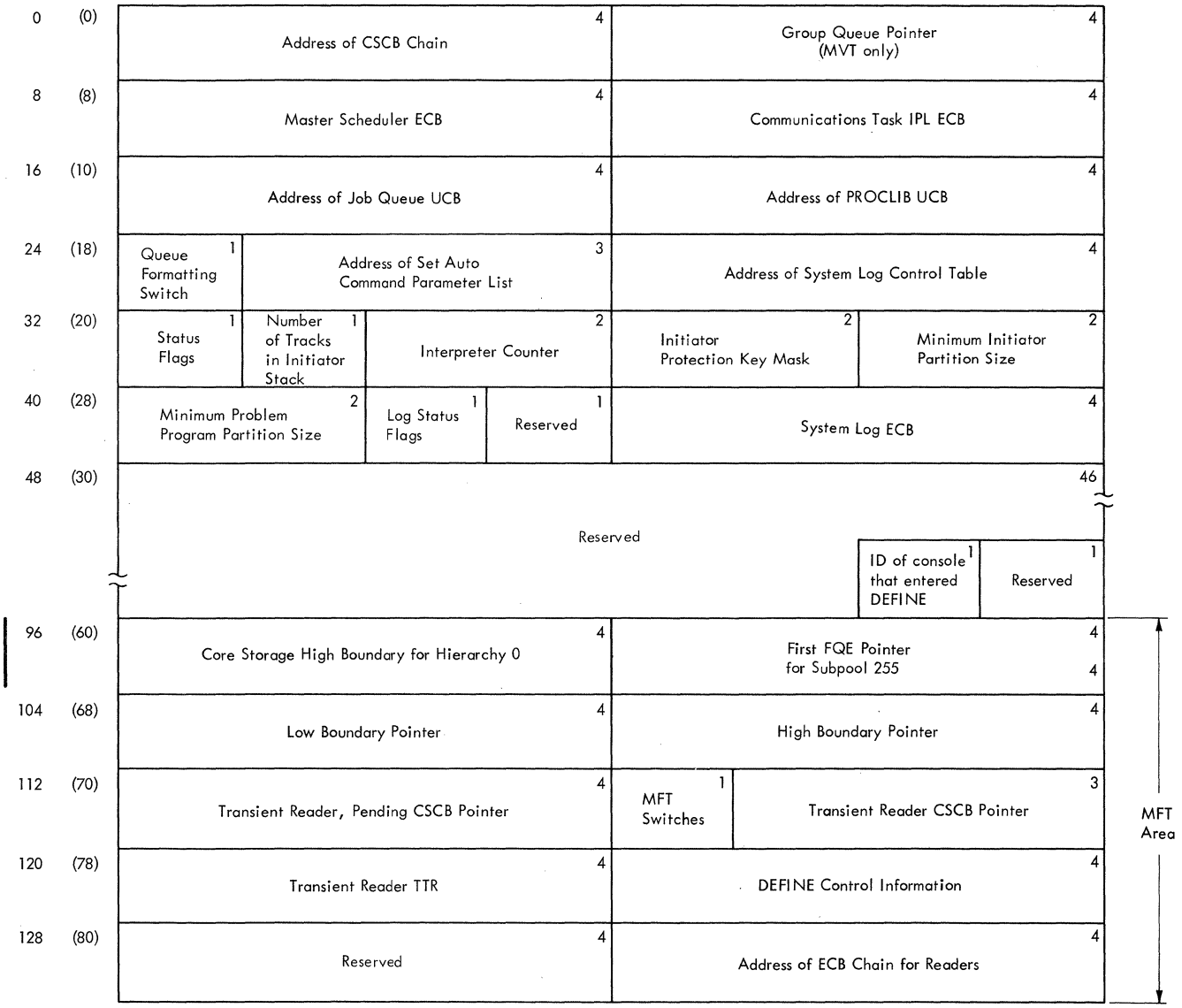


Figure 28. Master Scheduler Resident Data Area (Part 1 of 2)

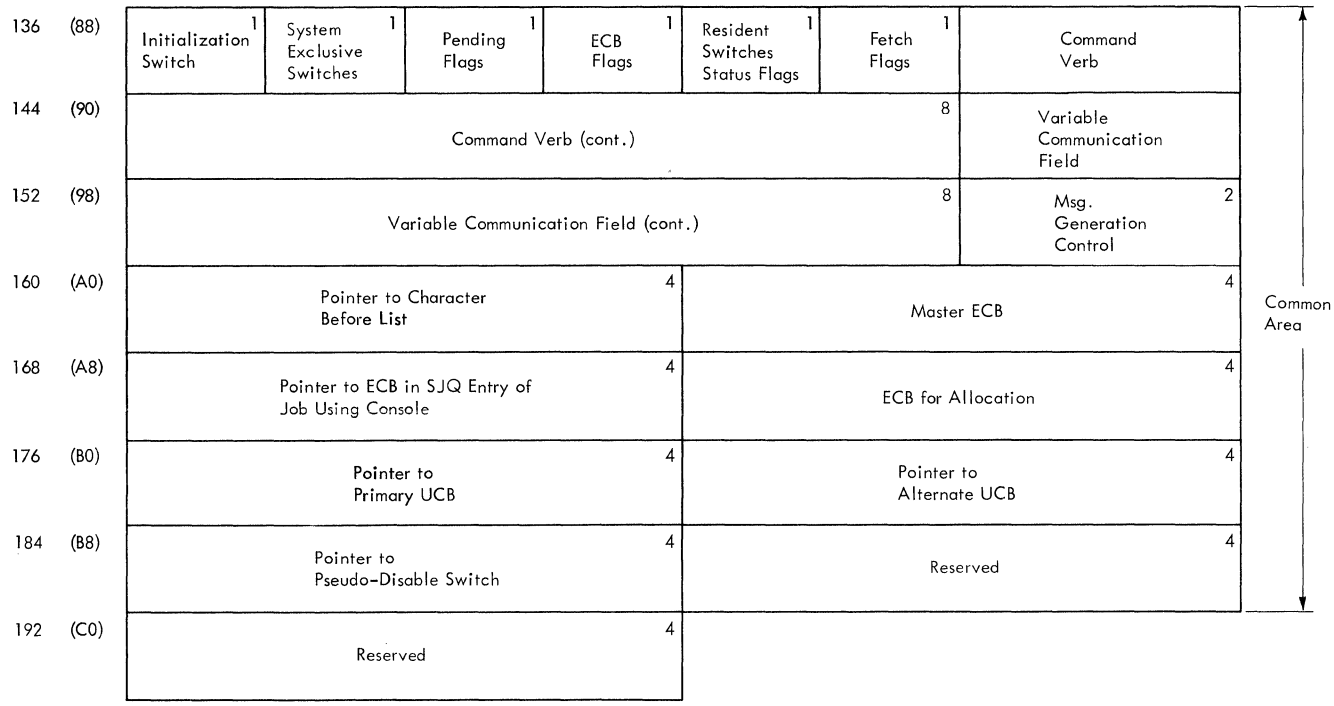


Figure 28. Master Scheduler Resident Data Area (Part 2 of 2)

PARTITION INFORMATION BLOCK

The 48-byte partition information block (PIB) (Figure 29) contains information used by the command processing and scheduler routines. Its location is stored in the TCBPIB field at displacement 124 (decimal) of the task control block (TCB).

Although most of the fields in the partition information block are self-explanatory, the following require further description:

- ECB Address: Contains the address of ECB to be posted by job selection when the partition is made quiescent for partition redefinition.

- "No Work" ECB for the Initiator: This ECB is posted by small partitions requesting service, the queue manager when a job has been enqueued, and by the DEFINE and START command routines.

- Status A Information:

Bit	Setting	Meaning
0	0	Stop initiator
	1	START INIT issued
1	1	Partition active
2	1	Reserved
3	1	Transient reader is suspended
4	1	Partition is to be terminated by IEFSD599 when it next gets control
5	1	Partition is involved in redefinition
6	1	System-assigned transient reader operating in this partition
7	1	Problem program is running

- Status B Information:

Bit	Setting	Meaning
0	1	Logical tracks added for initiator
1	1	LOT block exits
2	1	SPIL has been created
3	1	Reserved
4	1	Unending task present in partition
5	1	JOBLIB Switch
6	1	STEPLIB Switch
7	1	FETCHLIB Switch

- SPIL Address: The small partition information list (SPIL) is applicable to large partitions only.

- Job Class Codes: Contains one to three codes for the partition, arranged in descending numerical order, i.e., GRP3 is in the second byte of the field, followed by GRP2 and GRP1. The first byte contains the protection key for the partition, if the system has the storage protection feature.

- Internal Queue Status Bits:

Bit	Setting	Meaning
0	1	A large partition in which the DSDR processing step for a small partition (less than 12K) is to be executed
1	1	Reserved
2	1	A DEFINE command has been received and the partition is processing jobs on its internal queue.
3-7		Reserved

- Job Step Timing Status Bits:

Bit	Setting	Meaning
0	1	The job step TQE is being used for job step timing.
1	1	Indicates to the Initiator that the step being terminated was timed.
2-7		Reserved.

0	(0)	CSCB Address of Pending Command	4
4	(4)	ECB Address	4
8	(8)	"No Work" ECB for the Initiator	4
12	(C)	Status Bits - A (1)	Address of Current Job Step CSCB (3)
16	(10)	Status Bits - B (1)	SPIL Address (3)
20	(14)	CSCB Address of Current Task in Partition	4
24	(18)	Protection Key (1)	Job Class Codes (3)
28	(1C)	CSCB Address of Suspended Reader	4
32	(20)	Address of the Direct SYSOUT Control Block (DSOCB) Chain	4
36	(24)	Internal Queue Status Bits (1)	Address of Internal Queue of Job Names to be Restarted (3)
40	(28)	Job Step Timing Status Bits (1)	Address of the Job Step TQE (3)
44	(2C)	Count of Active Subtasks (1)	Address of the RB of the Most Recently Loaded Module on the JPAQ (3)

Figure 29. Partition Information Block (PIB)

SMALL PARTITION INFORMATION LIST (SPIL)

Description: The 80-byte small partition information list (SPIL) (Figure 30) is a storage area for information pertaining to small partition scheduling. It is built in main storage obtained from subpool 0. The address of the ECBs provides for information to be passed between the small partition and the large partition that is performing initiation, allocation, or termination functions for the small partition.

Most of the fields in the small partition information block are self explanatory; however, the status bits field is described below.

Bits 0 and 1 contain ones if a START writer or reader command has been entered.

Bit 2 contains a one if a SPIL pointer has been stored in the PIB.

Bit 3 contains a one if a problem program has requested termination.

Bit 4 contains a one if an indicative dump was requested.

Bits 0-7 contain zeros if a START INIT command was entered.

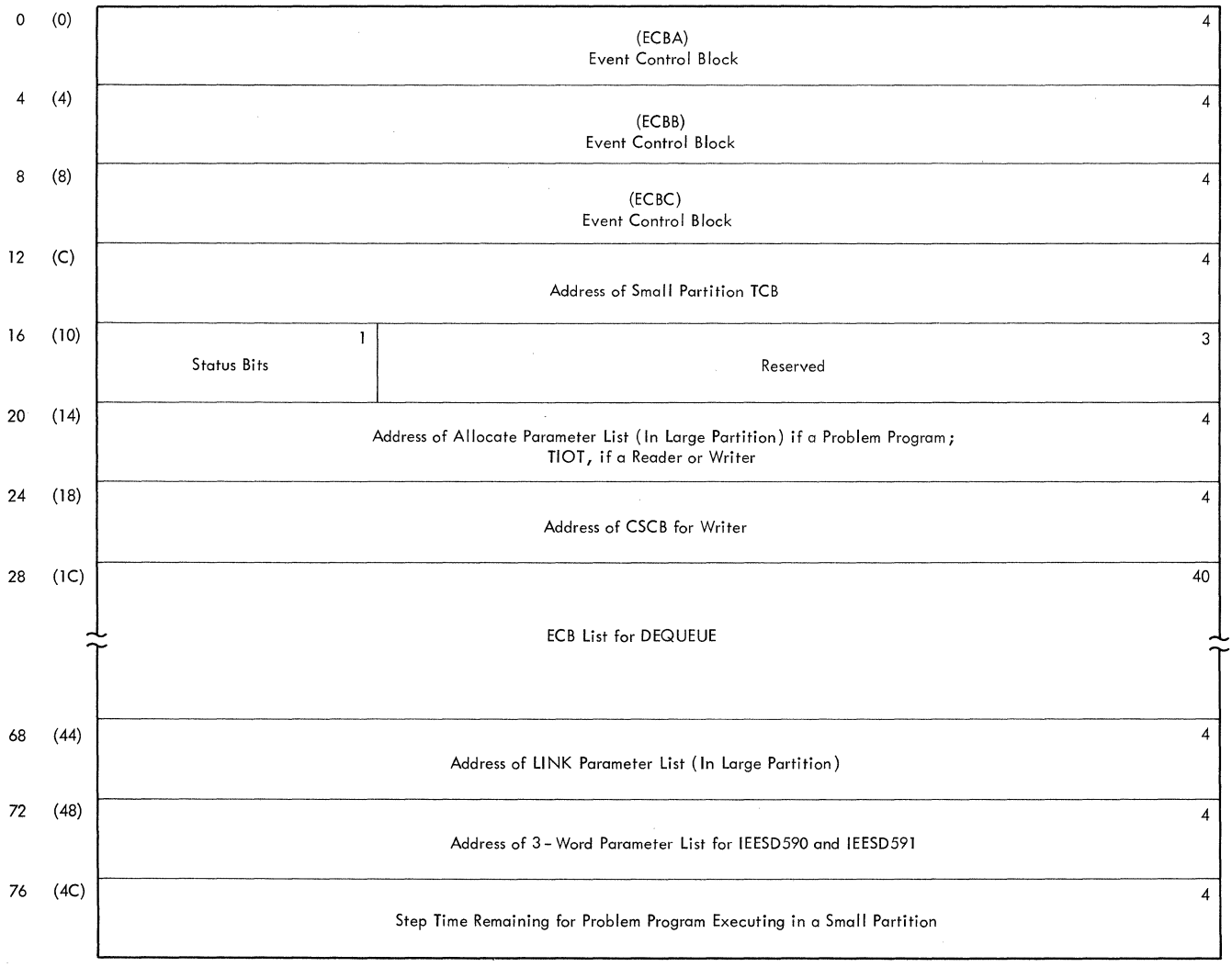


Figure 30. Small Partition Information List (SPIL)

STEP CONTROL TABLE (SCT)

Description: The step control table (SCT) (Figure 31), is used to pass control information to the DD routine of the interpreter and to the initiator routines, which also contribute information to the table. This table is created and initialized by the execute statement processor routine of the interpreter when an EXEC statement is read. One SCT is created for each step of a job.

If the step is part of a previously cataloged procedure, the name of the step that called the procedure, if any, is entered. The following variable-content and indicator fields are included in the table:

BYTE 4: Internal Step Status Indicators:

Bit	Setting	Meaning
0	1	Step can be rolled out
1	1	Roll step out if necessary
2	1	Do not restart step
3	1	Do not take a checkpoint
4	1	Restart if necessary
5	1	Graphics - alter protect key
6	1	Graphics - ABEND exit
7	1	Step failed

PARM Count or Step Status Code:

a. Interpreter: The number of characters specified in the PARM parameter of the EXEC statement is placed in this entry.

b. Initiator: This table entry contains the condition code returned by the processing program.

BYTE 67: Step Type Indicators:

Bit	Setting	Meaning
0	1	EXEC statement contains PGM=*.stepname.ddname
1	1	SYSIN is specified as DD*
2	1	SYSOUT is specified
3	1	JFCB housekeeping is complete
4-6		Initiator Indicator
7		Reserved

BYTE 104: Extension of Internal Step Status Indicator

Bit	Setting	Meaning
0		Reserved
1	1	Direct system output facilities required to output job separator or system messages.
2	1	Allocation for control volume
3		Reserved
4	1	STEPLIB present
5	1	Spooled SYSIN for step
6	1	Job ended
7		Reserved

Mapping Macro Instruction: IEFASCTB

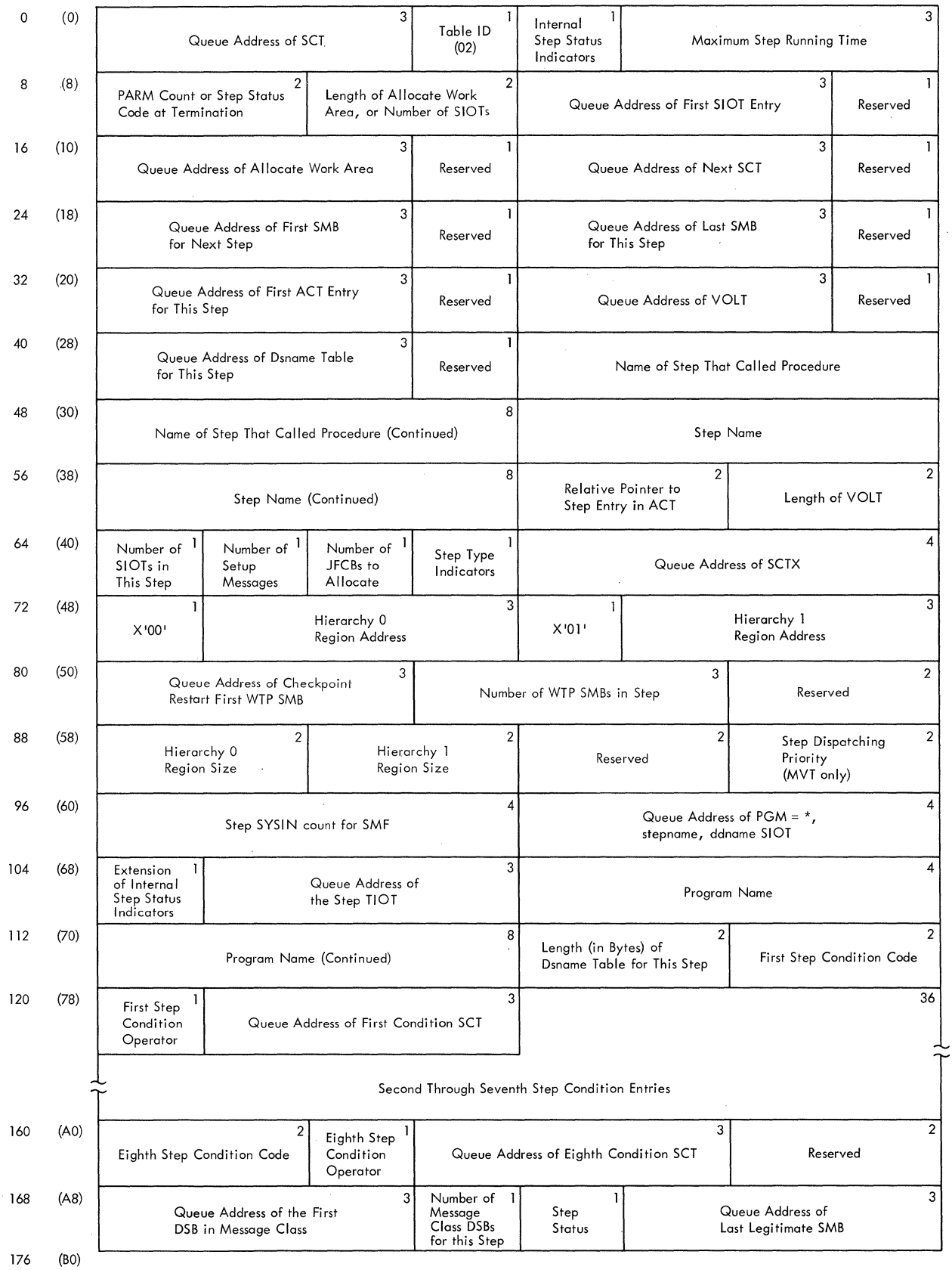


Figure 31. Step Control Table (SCT)

STEP INPUT/OUTPUT TABLE (SIOT)

Description: The Step Input/Output Table (SIOT) (Figure 32), makes DD statement available to the initiator for use as a source of information for the TIOT and for providing DD information to allocation and disposition routines. When a DD statement is read, the interpreter creates a new SIOT and places the DD information into it. The individual bits of the disposition byte and of indicator bytes 56 through 59 in the SIOT are set to one to indicate the following conditions:

BYTE 55: Scheduler Disposition

Bit	Meaning
0	Reserved
1	Retain volume
2	Private volume
3	Pass data set
4	Keep data set
5	Delete data set
6	Catalog data set
7	Uncatalog data set

BYTE 56: Indicator Byte Number 1

Bit	Meaning
0	Dummy data set
1	SYSIN data set
2	Split (primary)
3	Split (secondary)
4	Suballocate
5	Parallel mount
6	Unit affinity
7	Unit separation

BYTE 57: Indicator Byte Number 2

Bit	Meaning
0	Channel affinity
1	Channel separation
2	Volume affinity
3	JOBLIB DD statement
4	Unlabeled (no labels)
5	Pool DD statement
6	Defer mounting
7	Received data set

BYTE 58: Indicator Byte Number 3

Bit	Meaning
0	Volume reference
1	SYSIN expected (procedures only)
2	Allocate work table volume block indicator
3	Volume reference in step
4	SYSOUT was specified
5	NEW data set
6	MOD data set
7	OLD or SHR data set

BYTE 59: Indicator Byte Number 4

Bit	Meaning
0	Set by reader to indicate GDG single
1	Set by initiator to indicate GDG all
2	Volume serial number was found in passed data set queue (PDQ)
3	American National Standard label
4	Step processed
5	Intra-step volume affinity
6	Data set is in PDQ
7	1 = old or modified data set 0 = new data set

BYTE 92: Conditional Disposition

Bit	Meaning
0-3	Reserved
4	Keep data set
5	Delete data set
6	Catalog data set
7	Uncatalog data set

Mapping Macro Instruction: IEFASIOT

TASK INPUT/OUTPUT TABLE (TIOT)

Description: The Task Input/Output Table (TIOT) (Figure 33) provides data management routines with the addresses of the JFCBs and devices allocated to the data sets in a job step or system task. It is constructed by the I/O device allocation routine in main storage. The allocation routine also places a copy of the TIOT on the appropriate job class queue with the other tables for the job step. After the step completes processing, the TIOT is brought in from the job queue and placed in the upper portion of the partition. The step is then terminated, and the TIOT is deleted.

For further information on the TIOT, see IBM System/360 Operating System: System Control Blocks, GC28-6628.

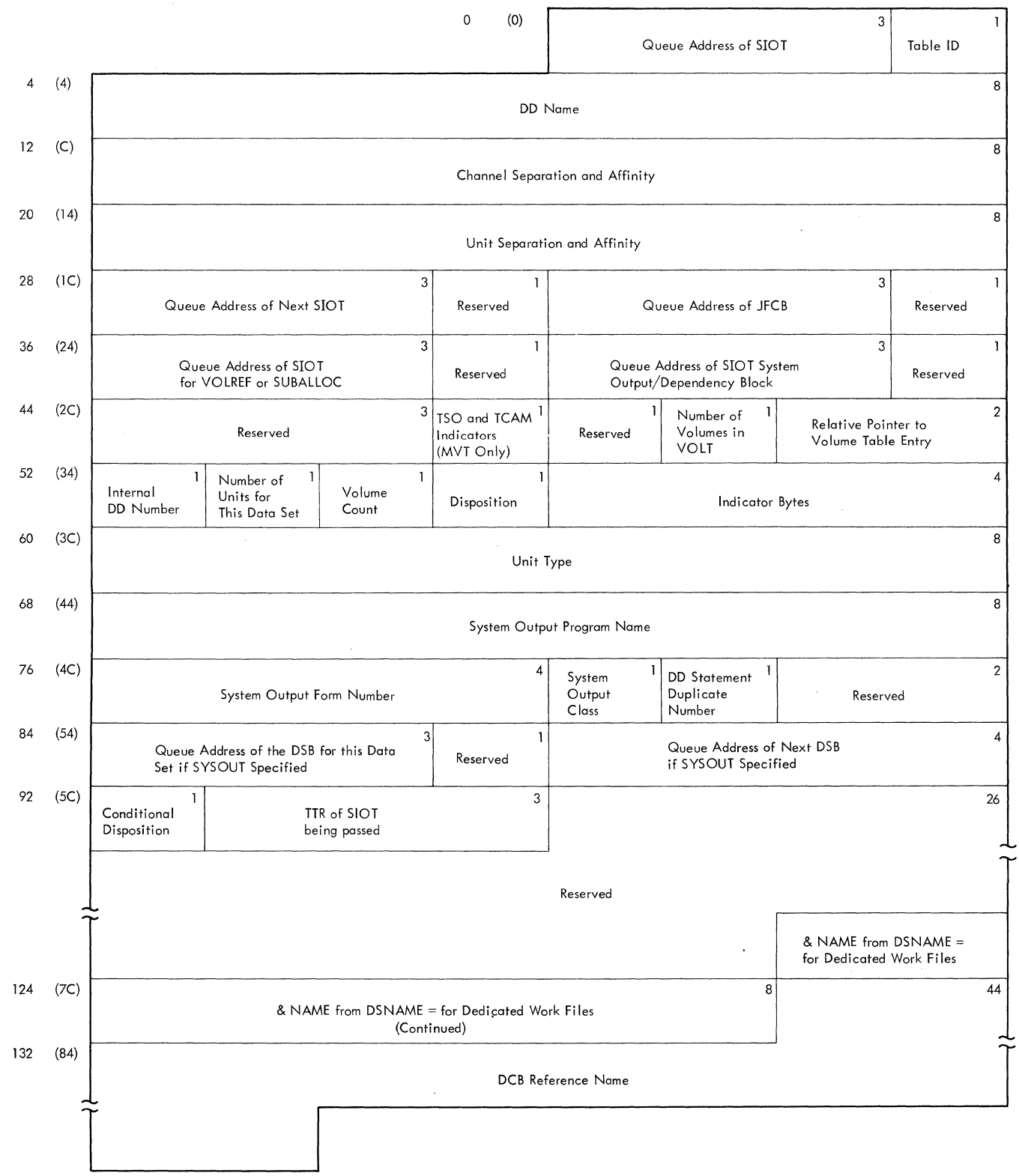


Figure 32. Step Input/Output Table (SIOT)

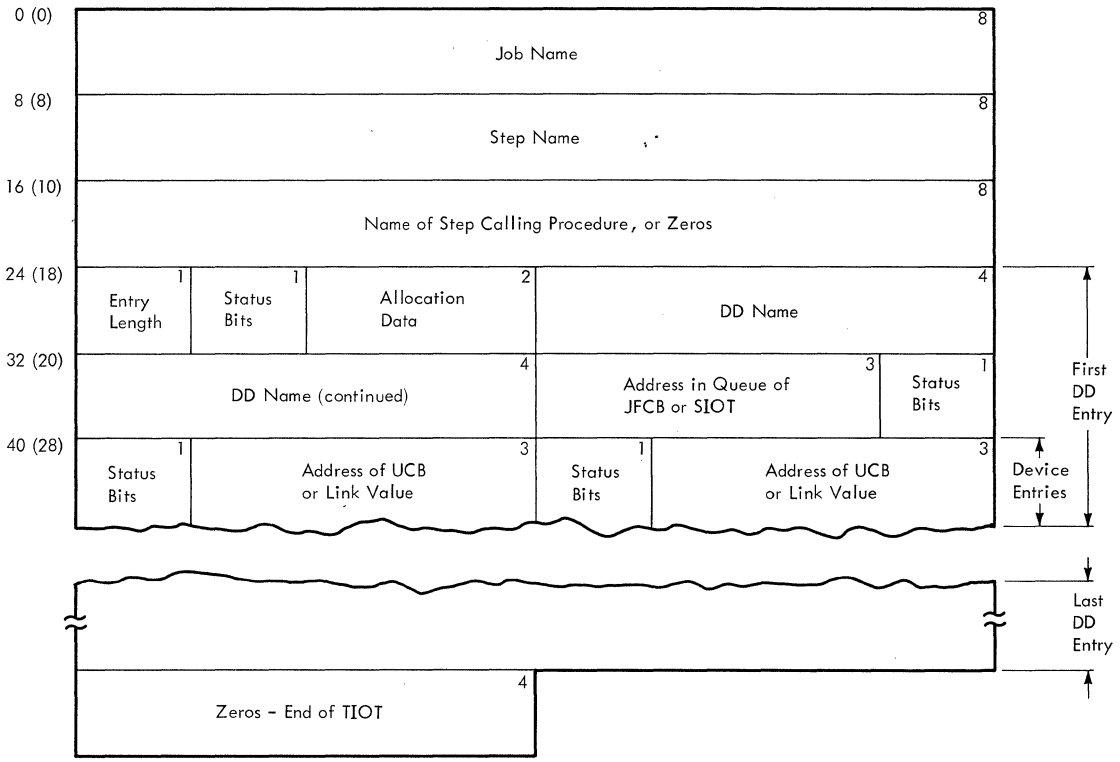


Figure 33. Task Input/Output Table (TIOT)

WRITE-TO-PROGRAMMER CONTROL BLOCK (WTPCB)

Description: The write-to-programmer control block (WTPCB) (Figure 34) is built by allocation interface control routine IEEVACTL in the system queue area. It is used by system tasks and problem program tasks when write-to-programmer messages are issued. The "Flags" field is defined below:

Bit	Setting	Meaning
0	1	Job queue problem
1	1	Limit message processed
2	1	Step contains SYSOUT
3	1	Return to IEFWTP01 upon completion
4	1	No record message processed
5	1	Last SMB used for job
6	1	WTP invoked for this step
7	1	WTOR or WTO with additional routing codes being processed by WTP

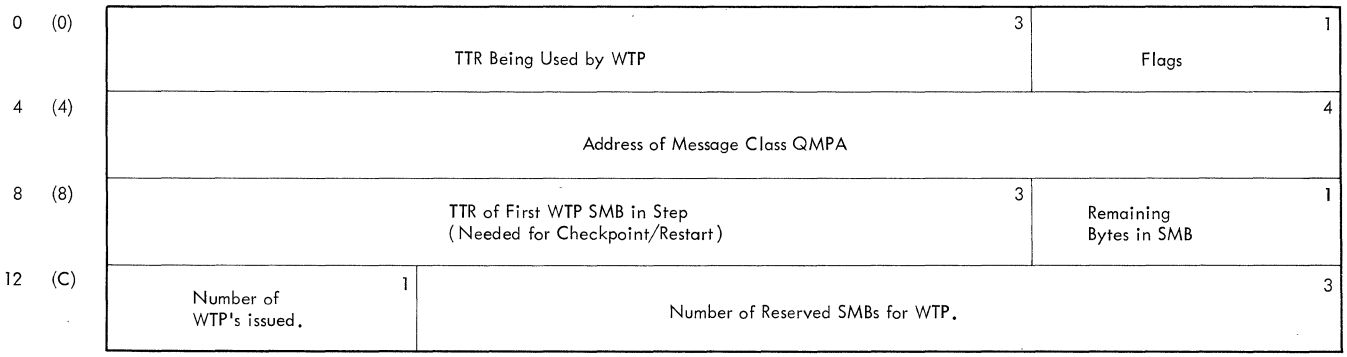


Figure 34. Write-to-Programmer Control Block (WTPCB)

APPENDIX B: MFT MODULES

This appendix contains a table of unique MFT job management modules, a group of tables showing the modules of each major component, a list matching entry point and control section names with source module names, and a brief description of each of the modules used by the MFT level of job management. If you are looking for a specific module and know only the major component and routine name, use Figures 36-46 which give a cross-reference to the source module. The source modules are in turn listed alphamerically for easy access. If you know the source module name, go directly to the module descriptions.

Unique MFT Modules

Figure 35 lists all modules that are unique to the MFT level of job management. This table is organized by major component.

Initiator:

IEFPPGM
IEFSD167
IEFSD32Q
IEFSD33Q
IEFSD510
IEFSD511
IEFSD512
IEFSD513
IEFSD515
IEFSD516
IEFSD517
IEFSD518
IEFSD519
IEFSD540
IEFSD541
IEFSD553
IEFSD554
IEFSD555
IEFSD556
IEFSD558
IEFSD559
IEFSD598
IEFSD599

I/O Device Allocation:

IEFSD551
IEFSD552
IEFSD557
IEF41DUM

Master Scheduler Task:

IEECIR50
IEECIR51
IEEDFINA
IEEDFINB
IFEDFINC
IEEDFIN1
IEEDFIN2
IEEDFIN3
IEEDFIN4
IEEDFIN5
IEEDFIN6
IEEDFIN7
IEEDFIN8
IEEDFIN9
IEESD566
IEFSD569

Nucleus:

IEESD568
IEFSD567

Queue Management:

IEFSD572

Reader/Interpreter:

IEFSD530
IEFSD531
IEFSD532
IEFSD533
IEFSD536
IEFSD537

SVC 34:

IEESD561
IEESD571
IEESD671
IEE2803D
IEE3903D
IEE7903D

System Management Facility:

IEESMFWT

System Task Control:

IEESD590
IEESD591
IEESD592
IEEVACTL
IEEVOMSG
IEEVSMBA
IEEVTCTL
IEFSD534
IEFSD535
IEFSD584
IEFSD585
IEFSD586
IEFSD587
IEFSD588
IEFSD589
IEF589SP

Figure 35. MFT Modules

Major Component Modules

Figures 36 through 46 list all MFT job management modules according to major component. The figures appear in alphabetical order by component name. Within each component, routine names are listed alphabetically with a cross-reference to the module name.

Routine	Source Module
Initialization	IEFDSOCP
Release DSOCB Routine	IEFDSOFB
SIOT and JFCB Modification	IEFDSOAL
STOP and MODIFY Command Processor	IEFDSOSM
System Messages and Job Separator Writer	IEFDSOWR
Tape to Printer or Card Punch	IEFPRINT

Figure 36. Direct System Output Modules

Routine	Source Module
Alternate Step Deletion	IEFSD516
Data Set Integrity	IEFSD541
Dequeue by Jobname Interface	IEFSD519
Dummy User Job Initiation Exit Routine	IEFUJI
Dummy User Step Initiation Exit Routine	IEFUSI
ENQ/DEQ Purge	IEFSD598
Job Deletion	IEFSD517
Job Initiation	IEFSD511
Job Selection	IEFSD510
Job Suspension	IEFSD168
Linkage from Job Termination to Initiator for the 30K Scheduler	IEFSD33Q
Linkage from Job Termination to Initiator for the 44K Scheduler	IEFSD32Q
Linkage to IEFSD168	IEFSD167
Linkage to IEFSD510	IEFSD555
Linkage to IEFSD511	IEFSD558
Linkage to IEFSD512	IEFSD553
Linkage to IEFSD515	IEFSD559
Linkage to IEFSD516	IEFSD554
Linkage to IEFSD541	IEFSD540
Partition Recovery	IEFSD518
Problem Program Initialization	IEFPPGM
Problem Program Interface	IEFSD513
Problem Program Table	IEFSDPPT
Set Problem Program State	IEFSD556
Shared DASD ENQ/DEQ Purge	IEFSD597
Small Partition Module	IEFSD599
Step Deletion	IEFSD515
Step Initiation	IEFSD512
TCTIOT Construction	IEFSMFAT
User Exit Initialization Routine	IEFSMFIE

Figure 37. Initiator Modules

Routine	Source Module
Allocation Control	IEFXCSSS
Allocation Entry	IEFSD21Q
Allocation Exit	IEFSD41Q
Allocation Recovery	IEFXJIMP
Allocation Recovery Messages	IEFSJMSG
Automatic Volume Recognition	IEFXV001
Automatic Volume Recognition Messages	IEFXVMSG
Automatic Volume Recognition Non-Standard Label Routine	IEFXVNSL
Automatic Volume Recognition Label Processing	IEFXV002
Automatic Volume Recognition Tape Processing	IEFXV003
Bit Pattern Scan Routine	IEFSCAN
DADSM Error Recovery	IEFXT003
Decision Allocation	IEFS5000
Demand Allocation	IEFWA000
Device Bit Pattern	IEFDEVPT
Device End Interrupt Handler	IEFSD567
Device Strikeout	IEFX300A
EXEC Statement Condition Code Processor	IEFVKIMP
EXEC Statement Condition Code Processor Messages	IEFVKMSG
Extended External Action	IEFWEXTA
External Action	IEFWD000
External Action Messages	IEFWD001
Interface	IEFSD557
JFCB Housekeeping Control and Allocate Processing	IEFVMLS1
JFCB Housekeeping Error Message Processing	IEFVMLS6
JFCB Housekeeping Error Messages	IEFVMLS7
JFCB Housekeeping Fetch DCB Processing	IEFVM2LS
JFCB Housekeeping GDG All Processing	IEFVM4LS
JFCB Housekeeping GDG Single Processing	IEFVM3LS
JFCB Housekeeping Patterning DSCB	IEFVM5LS
JFCB Housekeeping Unique Volume ID	IEFVM76
Linkage Module	IEFWCFAK
Linkage Module	IEFWSWIN
Linkage Module	IEFXJFAK
Linkage Module	IEF41FAK
Linkage to JFCB Housekeeping	IEFVMMS1
Linkage to IEFVMSL1	IEFVMFAK
Linkage to IEFXJIMP	IEFSD551
Linkage to IEFXJIMP	IEFSD552

Figure 38. I/O Device Allocation Modules (Part 1 of 2)

Routine	Source Module
Linkage to IEFXV001	IEFAVFAK
Linkage to Mount Control Volume	IEFCVFAK
Message Module	IEFK1MSG
Message Module	IEFWSTRT
Message Module	IEFXAMSG
Mount Control Volume	IEFMCVOL
Non-Recovery Error	IEFXKIMP
Non-Recovery Error Messages	IEFXKMSG
Return to Initiator or System Task Control	IEF41DUM
Separation Strikeout	IEFXH000
Space Request	IEFXT00D
VARY Interface and TIOT Compression	IEFXT002
TIOT Construction	IEFWCIMP
Volume Attribute Setting	IEFPRES
Wait for Space Decision	IEFSD097
Wait for Unallocation	IEFSD195

Figure 38. I/O Device Allocation Modules (Part 2 of 2)

Routine	Source Module
Command Statement	IEFVHM
CPO Allocation Subroutine	IEFVSD12
CPO	IEFVHG
Continuation Statement	IEFVHC
DD* Statement Generator	IEFVHB
DD Statement Processor	IEEFVDA
Data Set Name Table Construction	IEFVDBSD
Dictionary Entry	IEFVGI
Dictionary Search	IEFVGS
Dummy User JCL Validation Exit Routine	IEFUJV
End-of-File	IEFVHAA
EXEC Statement Processor	IEFVEA
Get Parameter	IEFVGK
Get	IEFVHA
Housekeeping	IEFVHHB
In-stream Procedure Compress Routine	IEZNCODE
In-Stream Procedure Directory Build Routine	IEFVINC
In-Stream Procedure Expand Routine	IEZDCODE
In-Stream Procedure Expand Interface Routine	IEFVIND

Figure 39. Interpreter Modules (Part 1 of 2)

Routine	Source Module
In-Stream Procedure Processor	IEFVINA
In-Stream Procedure Search Routine	IEFVINB
In-Stream Procedure Syntax Check Routine	IEFVINE
Initialization	IEFVH1
Initialization	IEFVH2
Interface	IEFSD533
Job and Step Enqueue	IEFVHH
Job Statement Processor	IEFVHA
Job Validity Check	IEFVHEC
Linkage Module	IEFSD537
Message Module	IEFVGM1
Message Module	IEFVGM2
Message Module	IEFVGM3
Message Module	IEFVGM4
Message Module	IEFVGM5
Message Module	IEFVGM6
Message Module	IEFVGM7
Message Module	IEFVGM8
Message Module	IEFVGM9
Message Module	IEFVGM10
Message Module	IEFVGM11
Message Module	IEFVGM12
Message Module	IEFVGM13
Message Module	IEFVGM14
Message Module	IEFVGM15
Message Module	IEFVGM16
Message Module	IEFVGM17
Message Module	IEFVGM18
Message Module	IEFVGM19
Message Module	IEFVGM70
Message Module	IEFVGM71
Message Module	IEFVGM78
Message Processing	IEFVGM
Null Statement	IEFVHL
Operator Message	IEFSD536
Post-Scan	IEFVHF
Pre-Scan Preparation	IEFVHEB
Queue Management Interface	IEFVHQ
Router	IEFVHE
Scan	IEFVFA
SCD Construction	IEFVSD13
Symbolic Parameter Processing	IEFVFB
Termination	IEFVHN
Test and Store	IEFVGT
Transient Reader Restore	IEFSD531
Transient Reader Suspend	IEFSD530
Transient Reader Suspend Tests	IEFSD532
Verb Identification	IEFVHCB

Figure 39. Interpreter Modules (Part 2 of 2)

Routine	Source Module
Command Analyzer Routine	IEECIR51
DEFINE Command Final Processor	IEEDFIN9
DEFINE Command Validity Check (Core Storage)	IEEDFINC
DEFINE Final Processor	IEEDFIN3
DEFINE Initialization	IEEDFIN1
DEFINE Keyword Scan	IEEDFIN7
DEFINE Listing	IEEDFIN4
DEFINE Message	IEEDFIN5
DEFINE Syntax Check and Router	IEEDFIN2
DEFINE System Reinitialization (1)	IEEDFIN8
DEFINE System Reinitialization (2)	IEEDFINB
DEFINE Time-Slice Syntax Check	IEEDFIN6
DISPLAY A	IEESD566
DISPLAY CONSOLES	IEEXEDNA
DISPLAY C,K, (1)	IEE10110
DISPLAY C,K, (2)	IEE11110
DISPLAY C,K, (3)	IEE12110
DISPLAY U (1)	IEE20110
DISPLAY U (2)	IEE21110
DISPLAY U (3)	IEE22110
DISPLAY U (4)	IEE23110
DISPLAY PFK	IEE40110
DQ/DN Message Setup	IEESD584
DUMP	IEE60110
ECB/IOB Construction	IEESD582
Log Open Initialization	IEELOG02
Log WAIT and Writer	IEELWAIT
Master Scheduler Initialization	IEFSD569
Master Scheduler Resident Data Area	IEESD568
Master Scheduler Router	IEE00110
Message Module	IEESD580
Public/Private Interface	IEEVPRES
Queue Alter Delete	IEESD576
Queue Message Class Set-Up	IEESD578
Queue Restart Enqueue	IEESD577
Queue Scratch	IEESD581
Queue Scratch Set-Up	IEESD575
Queue Search	IEESD564
Queue Search Return	IEESD583
Queue Search Set-Up	IEESD563
Queue SMB Routine	IEESD579
Resident Command Processor	IEECIR50
Service	IEESD565
SMF Initialization (1)	IEESMFIT
SMF Initialization (2)	IEESMFI2
SMF MFT Storage Configuration Record Creation	IEEDFINA
SMF Open Initializer	IEESMFOI

Figure 40. Master Scheduler Modules (Part 1 of 2)

Routine	Source Module
SMF Parameter Processor	IEESMFI3
System Log Data Set Initialization	IEEVLOUT
System Log Dispatcher	IEEVLDSP
System Log Initialization	IEEVLIN
System Log Open Initializer	IEEVLIN2
System Log SVC (SVC 36)	IEE0303F
System Log SVC (SVC 36 - second load)	IEE0403F
Syntax Check	IEESD562
Figure Look-Up Routine	IEEVRFRX
Write-to-Programmer Error Processing	IEFWTP02
Write-to-Programmer Initialization	IEFWTP00
Write-to-Programmer Message Processor	IEFWTP01

Figure 40. Master Scheduler Modules (Part 2 of 2)

Routine	Source Module
Assign	IEFQASGQ
Assign/Start	IEFQAGST
Branch	IEFQMLK1
Control	IEFQBVMS
Delete	IEFQDELQ
Dequeue	IEFQMDQQ
Dequeue by Jobname	IEFLOCDQ
Dequeue by Jobname Interface	IEFSD519
Dummy	IEFQMDUM
Enqueue	IEFQMNQQ
Interpreter/Queue Manager Interlock	IEFSD572
Message Module	IEFSD311
Queue Formatting	IEFORMAT
Queue Initialization	IEFSD055
Queue Manager Table Breakup	IEFSD514
Read/Write	IEFQMRAW
Resident Main Storage Reservation	IEFQRESD
Transient Queue Manager Initialization	IEFXQM00
Transient Queue Manager Record Assignment	IEFXQM02
Transient Queue Manager Track Assignment	IEFXQM01
Unchain	IEFQMUNQ

Figure 41. Queue Management Modules

Routine	Source Module
CANCEL Processor	IEE2803D
Command Rejector	IKJNULL
Command Translator	IEE5403D
CONTROL Command Handler (1)	IEE6703D
CONTROL Command Handler (2)	IEE6703D
CONTROL Command Handler (3)	IEE7703D
CONTROL Command Handler (4)	IEE7803D
CONTROL Command Handler (5)	IEE6903D
CSCB and CIB Chain Manipulator	IEE0303D
CSCB Creation	IEE0803D
DEFINE, MOUNT Routine	IEESD571
DISPLAY Processor	IEE3503D
DISPLAY Requests	IEE2903D
DISPLAY SQA	IEE8503D
HALT (EOD Routine)	IEE1403D
HARDCPY Message Routine	IEE4103D
LOG and WRITE LOG Routine	IEE1603D
Machine Status Control Model 85 (1)	IGF08501
Machine Status Control Model 85 (2)	IGF08502
Machine Status Control	
Machine Status Control Model 135	IGF13501
Machine Status Control Model 145	IGF29701
Machine Status Control Model 155	IGF29601
Machine Status Control Model 165	IGF55301
Master Command EXCP Routine	IGC0103D
MCS Periodic STOP Handler (JOBNAMES, STATUS, DSNAME, SPACE)	IEE5503D
MCS REPLY Processor	IEE1A03D
REPLY Messages	IEE1B03D
MCS VARY Syntax Check	IEE3303D
Message Assembly	IEE0503D
Message Assembly	IEE2103D
Message Module	IEE7903D
MODE Command Router	IGF2603D
MONITOR Command Router	IEE7103D
MSGRT Command Processor (1)	IEE6303D
MSGRT Command Processor (2)	IEE6403D
MSGRT and CONTROL Message Module (1)	IEE6503D

Figure 42. SVC Modules (Part 1 of 2)

Routine	Source Module
MSGRT and CONTROL Message Module (2)	IEE5903D
Periodic STOP Handler (JOBNAMES, STATUS, DSNAME, SPACE)	IEE4503D
REPLY Processor	IEE1203D
RJE Command Processor	IEE1503D
Router	IEE0403D
Routing Looation (I)	IEE7503D
Routing Location (II)	IEE7603D
SET Command Processor (Part I)	IEE0603D
SET Command Processor (Part II)	IEE8603D
SET TOD Clock (Part I)	IEE6503D
SET TOD Clock (Part II)	IEE6603D
START and STOP INIT Processor (1)	IEESD561
START and STOP INIT Processor (2)	IEE3903D
STOP and MODIFY Scheduling	IEE0703D
SWAP Command Processor	IGF2503D
SWITCH Command Processor	IEE1403D
System Management Facility VARY Record Handler	IEE2303D
Timer Maintenance	IEE0903D
VARY CONSOLE Information Message Routine (Load I)	IEE4803D
VARY CONSOLE Information Message Routine (Load II)	IEE7303D
VARY CONSOLE Keyword Scan	IEE4403D
VARY CONSOLE Processor	IEE4903D
VARY HARDCOPY Processor (Load I)	IEE4703D
VARY HARDCOPY Processor (Load II)	IEE7203D
VARY HARDCPY OFF Processor	IEE5703D
VARY Keyword Router	IEE3203D
VARY MSTCONS Processor	IEE4303D
VARY ONGFX/OFFGFX Handler	IEE1703D
VARY ONLINE/OFFLINE Router	IEE4603D
VARY PATH Command Processor	IGF2403D
VARY Secondary Scan	IEE4203D
VARY and UNLOAD Scan for Non-MCS Systems	IEE1103D
VARY and UNLOAD Processor	IEE3103D

Figure 42. SVC 34 Modules (Part 2 of 2)

Routine	Source Module
Class Name Setup	IEFSD081
Command Chaining Access Method	IEFSDXYZ
Command Processing	IEFSD083
Data Set Delete	IEFSD171
Data Set Writer Interface	IEFSD070
DSB Handler	IEFSD085
Initialization	IEFSD080
Job Separator	IEFSD094
Linkage Module	IEF078SD
Linkage Module	IEF079SD
Linkage Module	IEF082SD
Linkage Module	IEF083SD
Linker	IEFSD078
Linkage to Queue Manager Delete	IEFSD079
Main Logic	IEFSD082
Message Module	IEFSD096
Print Line	IEFSD095
Put	IEFSD089
SMB Handler	IEFSD086
Spanned Data Sets	IEFSDXXX
Standard Writer	IEFSD087
Transition	IEFSD088
Wait	IEFSD084

Figure 43. System Output Writer Modules

Routine	Source Module
Initialization	IEFSD300
Jobnames Figure	IEFSD302
Linkage Module	IEF300SD
Linkage Module	IEF304SD
Message Module	IEFSD312
Purge Queue Construction	IEFSD301
Reenqueue	IEFSD305
Scratch Data Sets	IEFSD304
Scratch Data Sets	IEFSD308
TTR and NN to MBBCCHHR Conversion	IEFSD310

Figure 44. System Restart Modules

Routine	Source Module
Allocation Interface	IEEVACTL
Internal JCL Reader	IEEVRJCL
Interpreter Exit	IEEVRC
Interpreter Post Scan Exit	IEEPSN
JCL Edit	IEEVJCL
Linkage to IEE591SD	IEFSD588
Linkage to IEFSD534	IEFSD589
Linkage to IEFSD535	IEFSD587
Linkage to IEFSD584	IEF589SP
Linkage to IEFSD585	IEFSD586
Linker	IEESD591
Link-Table	IEEVLNKT
LPSW	IEFSD534
LPSW	IEFSD535
LPSW	IEFSD584
LPSW	IEFSD585
Message Writer	IEEVSMSG
Message Writing	IEEVOMSG
POST	IEESD592
QMPA Builder	IEEVSMBA
Reader/Interpreter Control	IEEVRCTL
START Syntax Check	IEEVSTAR
Termination Interface	IEEVTCTL
Transient Reader Linker	IEESD591
Write TIOT on Disk	IEESD590

Figure 45. System Task Control Modules

Routine	Source Module
Disposition and Deallocation Messages	IEFZGMSG
VARY Interface and Disposition and Deallocation Messages	IEFZHMSG
Disposition and Deallocation	IEFZGJB1
Disposition and Deallocation	IEFZGST1
DSB Processing	IEFYTVMS
Dummy Accounting	IEFACTRT
Job Statement Condition Code Processor	IEFVJ1MP
Job Statement Condition Code Processor Messages	IEFVJMSG
Job Termination Control	IEFZAJB3
Job Termination Exit	IEFSD31Q
Message Blocking	IEFYSVMS
Message Module	IEFIDMPM
Message Module	IEFWTERM
Restart Preparation	IEFRPREP
SMF Writer Interface	IEFSMFWI
Step Termination Control	IEFYNIMP
Step Termination Control Routine Messages	IEFYNMSG
Step Termination Data Set Driver	IEFYPJB3
Step Terminate Exit	IEFSD22Q
Step Termination Messages	IEFYPMSG
System Output Interface	IEFSD017
Termination Entry	IEFSD42Q
Unallocation Routine	IEFZGSTZ
User Accounting Routine Linkage	IEFACTLK
User Dummy Accounting	IEFACTFK
User Exit Initialization	IEFSMFLK

Figure 46. Termination Modules

Module Cross Reference

This section contains an alphameric list of entry point and control section names, together with the name of the module that contains them. For further information on the modules, refer to the module descriptions.

Entry Point or Control Section Name	Module Name
GO	IEFSD515
IEAQOT00	IEE0903D
IEECIR50	IEECIR50
IEEDFINA	IEEDFINA
IEEDFINB	IEEDFINB
IEEDFINC	IEEDFINC
IEEDFIN1	IEEDFIN1
IEEDFIN2	IEEDFIN2
IEEDFIN3	IEEDFIN3
IEEDFIN4	IEEDFIN4
IEEDFIN5	IEEDFIN5
IEEDFIN6	IEEDFIN6
IEEDFIN7	IEEDFIN7
IEEDFIN8	IEEDFIN8
IEEDFIN9	IEEDFIN9
IEEDPART	IEEDFIN2
IEELOG02	IEELOG02
IEELWAIT	IEELWAIT
IEEMSER	IEESD568
IEEMSTWO	IEESD579
IEEPSN	IEEPSN
IEEREXIT	IEEDFIN9
IEESD562	IEESD562
IEESD563	IEESD563
IEESD564	IEESD564
IEESD565	IEESD565
IEESD566	IEESD566
IEESD575	IEESD575
IEESD576	IEESD576
IEESD577	IEESD577
IEESD578	IEESD578
IEESD579	IEESD579
IEESD580	IEESD580
IEESD581	IEESD581
IEESD582	IEESD582
IEESD583	IEESD583
IEESD584	IEESD584
IEESD590	IEESD590
IEESD591	IEESD591
IEESD592	IEESD592
IEESD82A	IEESD582
IEESMFAL	IEESMFAL
IEESMFIO	IEESMFI3
IEESMFIT	IEESMFIT
IEESMFI2	IEESMFI2

(continued)

Entry Point or Control Section Name	Module Name
IEESMFI3	IEESMFI3
IEESMFI4	IEESMFIT
IEESMFMS	IEESMFI3
IEESMFOI	IEESMFOI
IEESMFOP	IEESMFOP
IEESMFWT	IEESMFWT
IEEVACTL	IEEVACTL
IEEVICLR	IEEVICLR
IEEVJCL	IEEVJCL
IEEVLDSP	IEEVLDSP
IEEVLIN	IEEVLIN
IEEVLNKT	IEEVLNKT
IEEVACTL	IEEVACTL
IEEVRCTL	IEEVRCTL
IEEVLOUT	IEEVLOUT
IEEVOMSG	IEEVOMSG
IEEVPRES	IEEVPRES
IEEVRC	IEEVRC
IEEVRFRX	IEEVRFRX
IEEVRJCL	IEEVRJCL
IEEVSMBA	IEEVSMBA
IEEVSMSG	IEEVSMSG
IEEVSTAR	IEEVSTAR
IEEVTCTL	IEEVTCTL
IEEXEDNA	IEEXEDNA
IEE0303D	IEE0303D
IEE0303F	IEE0303F
IEE0403D	IEE0403D
IEE0403F	IEE0403F
IEE0503D	IEE0503D
IEE0603D	IEE0603D
IEE0703D	IEE0703D
IEE0803D	IEE0803D
IEE10030	IEE10030
IEE10110	IEE10110
IEE11110	IEE11110
IEE1103D	IEE1103D
IEE1203D	IEE1203D
IEE12110	IEE12110
IEE1403D	IEE1403D
IEE1603D	IEE1603D
IEE1703D	IEE1703D
IEE1A03D	IEE1A03D
IEE1B03D	IEE1B03D
IEE20110	IEE20110
IEE2103D	IEE2103D
IEE21110	IEE21110

(continued)

Entry Point or Control Section Name	Module Name
IEE22110	IEE22110
IEE2303D	IEE2303D
IEE23110	IEE23110
IEE2803D	IEE2803D
IEE2903D	IEE2903D
IEE3103D	IEE3103D
IEE3203D	IEE3203D
IEE3303D	IEE3303D
IEE3503D	IEE3503D
IEE3603D	IEE3603D
IEE3903D	IEE3903D
IEE40110	IEE40110
IEE4103D	IEE4103D
IEE4203D	IEE4203D
IEE4303D	IEE4303D
IEE4403D	IEE4403D
IEE4503D	IEE4503D
IEE4603D	IEE4603D
IEE4703D	IEE4703D
IEE4803D	IEE4803D
IEE4903D	IEE4903D
IEE5403D	IEE5403D
IEE5503D	IEE5503D
IEE5603D	IEE5603D
IEE5703D	IEE5703D
IEE5903D	IEE5903D
IEE591SD	IEESD591
IEE60110	IEE60110
IEE6303D	IEE6303D
IEE6403D	IEE6403D
IEE6503D	IEE6503D
IEE6603D	IEE6603D
IEE6703D	IEE670OD
IEE6803D	IEE6803D
IEE6903D	IEE6903D
IEE7103D	IEE7103D
IEE7203D	IEE7203D
IEE7303D	IEE7303D
IEE7503D	IEEM503D
IEE7603D	IEE7603D
IEE7703D	IEE7703D
IEE7803D	IEE7803D
IEE7903D	IEE7903D
IEE8503D	IEE8503D
IEE8603D	IEE8603D
IEFACTLK	IEFACTFK
IEFACTLK	IEFACTLK
IEFACTLK	IEFSMFLK
IEFACTRT	IEFACTRT
IEFALRET	IEFSD512
IEFCVOL1	IEFCVFAK

(continued)

Entry Point or Control Section Name	Module Name
IEFCVOL1	IEFMCVOL
IEFCVOL2	IEFCVFAK
IEFCVOL2	IEFMCVOL
IEFCVOL3	IEFCVFAK
IEFCVOL3	IEFMCVOL
IEFDPOST	IEFSD567
IEFDSDRP	IEFDSDRP
IEFDSOAL	IEFDSOAL
IEFDSOCP	IEFDSOCP
IEFDSOFB	IEFDSOFB
IEFDSOSL	IEFSD511
IEFDSOSM	IEFDSOSM
IEFDSOWR	IEFDSOWR
IEFIDMPM	IEFIDMPM
IEFIRC	IEFSD533
IEFJOB	IEFQRESD
IEFKG	IEFSD532
IEFK1MSG	IEFK1MSG
IEFORMAT	IEFORMAT
IEFPH2	IEFSD531
IEFPPGM	IEFPPGM
IEFPRES	IEFPRES
IEFQAGST	IEFQAGST
IEFQASNM	IEFQASGQ
IEFQDELE	IEFQDELQ
IEFQMDQ2	IEFQMDQQ
IEFQMDUM	IEFQMDUM
IEFQMNQ2	IEFQMNQQ
IEFQMRAW	IEFQMRAW
IEFQMSSS	IEFQBVMS
IEFQMSSS	IEFQMDUM
IEFQMSSS	IEFQMLK1
IEFQMUNC	IEFQMUNQ
IEFRCLN1	IEFRCLN1
IEFRCLN2	IEFRCLN2
IEFRPREP	IEFRPREP
IEFSDPPT	IEFSDPPT
IEFRSTRT	IEFRSTRT
IEFSDXXX	IEFSDXXX
IEFSDXYZ	IEFSDXYZ
IEFSD012	IEFVSD12
IEFSD017	IEFSD017
IEFSD055	IEFSD055
IEFSD068	IEFSD167
IEFSD068	IEFSD168
IEFSD070	IEFSD070
IEFSD071	IEFSD171
IEFSD078	IEFSD078
IEFSD078	IEF078SD
IEFSD079	IEFSD079
IFDSD079	IEF079SD
IEFSD080	IEFSD080
IEFSD081	IEFSD081

(continued)

Entry Point or Control Section Name	Module Name
IEFSD082	IEF082SD
IEFSD082	IEFSD082
IEFSD083	IEFSD083
IEFSD083	IEF083SD
IEFSD084	IEFSD084
IEFSD085	IEFSD085
IEFSD086	IEFSD086
IEFSD087	IEFSD087
IEFSD088	IEFSD088
IEFSD089	IEFSD089
IEFSD090	IEFVSD13
IEFSD094	IEFSD094
IEFSD095	IEFSD095
IEFSD095	IEFSD195
IEFSD096	IEFSD096
IEFSD097	IEFSD097
IEFSD300	IEFSD300
IEFSD300	IEF300SD
IEFSD301	IEFSD301
IEFSD302	IEFSD302
IEFSD303	IEFSD303
IEFSD304	IEFSD304
IEFSD304	IEF304SD
IEFSD305	IEFSD305
IEFSD308	IEFSD308
IEFSD310	IEFSD310
IEFSD311	IEFSD311
IEFSD312	IEFSD312
IEFSD41R	IEF41DUM
IEFSD510	IEFSD510
IEFSD511	IEFSD511
IEFSD512	IEFSD512
IEFSD512	IEFSD553
IEFSD513	IEFSD513
IEFSD514	IEFSD514
IEFSD515	IEFSD515
IEFSD516	IEFSD516
IEFSD517	IEFSD517
IEFSD518	IEFSD518
IEFSD519	IEFSD519
IEFSD530	IEFSD530
IEFSD531	IEFSD531
IEFSD534	IEFSD534
IEFSD535	IEFSD535
IEFSD537	IEFSD537
IEFSD540	IEFSD540
IEFSD541	IEFSD541
IEFSD554	IEFSD554
IEFSD555	IEFSD555
IEFSD556	IEFSD556
IEFSD557	IEFSD557
IEFSD558	IEFSD558
IEFSD559	IEFSD559

(continued)

Entry Point or Control Section Name	Module Name
IEFSD569	IEFSD569
IEFSD572	IEFSD572
IEFSD573	IEFSD572
IEFSD584	IEFSD584
IEFSD585	IEFSD585
IEFSD586	IEFSD586
IEFSD587	IEFSD587
IEFSD588	IEFSD588
IEFSD589	IEFSD589
IEFSD598	IEFSD597
IEFSD598	IEFSD598
IEFSD599	IEFSD599
IEFSD71M	IEFSD171
IEFSD83M	IEFSD083
IEFSD85M	IEFSD085
IEFSD86M	IEFSD086
IEFSD87M	IEFSD087
IEFSD89M	IEFSD089
IEFSMFAT	IEFSMFAT
IEFSMFIE	IEFSMFIE
IEFSMFWI	IEFSMFWI
IEFSMR	IEFRSTRT
IEFUCBL	IEFWA000
IEFUJI	IEFUJI
IEFUJV	IEFUJV
IEFUSI	IEFUSI
IEFVAWAT	IEFSD195
IEFVDA	IEFVDA
IEFVDBSD	IEFVDBSD
IEFVEA	IEFVEA
IEFVFA	IEFVFA
IEFVFB	IEFVFB
IEFVGI	IEFVGI
IEFVGK	IEFVGK
IEFVGM	IEFVGM
IEFVGM1	IEFVGM1
IEFVGM2	IEFVGM2
IEFVGM3	IEFVGM3
IEFVGM4	IEFVGM4
IEFVGM5	IEFVGM5
IEFVGM6	IEFVGM6
IEFVGM7	IEFVGM7
IEFVGM8	IEFVGM8
IEFVGM9	IEFVGM9
IEFVGM10	IEFVGM10
IEFVGM11	IEFVGM11
IEFVGM12	IEFVGM12
IEFVGM13	IEFVGM13
IEFVGM14	IEFVGM14
IEFVGM15	IEFVGM15
IEFVGM16	IEFVGM16
IEFVGM17	IEFVGM17
IEFVGM18	IEFVGM18
IEFVGM19	IEFVGM19

(continued)

Entry Point or Control Section Name	Module Name
IEFVGM70	IEFVGM70
IEFVGM71	IEFVGM71
IEFVGM78	IEFVGM78
IEFVGS	IEFVGS
IEFVGT	IEFVGT
IEFVHA	IEFVHA
IEFVHAA	IEFVHAA
IEFVHB	IEFVHB
IEFVHC	IEFVHC
IEFVHCB	IEFVHCB
IEFVHE	IEFVHE
IEFVHEB	IEFVHEB
IEFVHEC	IEFVHEC
IEFVHF	IEFVHF
IEFVHG	IEFVHG
IEFVHH	IEFVHH
IEFVHHB	IEFVHHB
IEFVHL	IEFVHL
IEFVHM	IEFVHM
IEFVHN	IEFVHN
IEFVHQ	IEFVHQ
IEFVHR	IEFSD536
IEFVH1	IEFVH1
IEFVH2	IEFVH2
IEFVINA	IEFVINA
IEFVINB	IEFVINB
IEFVINC	IEFVINC
IEFVIND	IEFVIND
IEFVINE	IEFVINE
IEFVJ	IEFVJIMP
IEFVJA	IEFVJA
IEFVJMSG	IEFVJMSG
IEFVK	IEFVKIMP
IEFVKMJ1	IEFVKMSG
IEFVKMSG	IEFVKMSG
IEFVM	IEFVMLS1
IEFVMCVL	IEFVMFAK
IEFVMCVL	IEFVMLS1
IEFVMQMI	IEFVMLS1
IEFVMSGR	IEFVMLS6
IEFVM1	IEFVMLS1
IEFVM1	IEFVMMS1
IEFVM2	IEFVM2LS
IEFVM3	IEFVM3LS
IEFVM4	IEFVM4LS
IEFVM5	IEFVM5LS
IEFVM6	IEFVMLS6
IEFVM7	IEFVMLS7
IEFVM76	IEFVM76
IEFVRRC	IEFVRRC
IEFVRRCA	IEFVRRC

(continued)

Entry Point or Control Section Name	Module Name
IEFVRRCB	IEFVRRC
IEFVRR1	IEFVRR1
IEFVRR2	IEFVRR2
IEFVRR3	IEFVRR3
IEFVSDRA	IEFVSDRA
IEFVSDRD	IEFVSDRD
IEFVSMBR	IEFVSMBR
IEFV15XL	IEFXJIMP
IEFV15XL	IEFSD551
IEFVR2AE	IEFVRR2
IEFVR3AE	IEFVRR3
IEFWA000	IEFWA000
IEFWA002	IEFWA000
IEFWA7	IEFWA000
IEFWC000	IEFWCFAK
IEFWC000	IEFWCIMP
IEFWC002	IEFWCFAK
IEFWD000	IEFWDFAK
IEFWD000	IEFWD000
IEFWD001	IEFWD001
IEFWD002	IEFWD000
IEFWDMSG	IEFWD000
IEFWEXTA	IEFWEXTA
IEFWLISD	IEFSD21Q
IEFWSMSG	IEFWSMSG
IEFWSTRT	IEFWSTRT
IEFWSWIT	IEFWSWIN
IEFWSYP3	IEFWMSG
IEFWTERM	IEFWTERM
IEFW1FAK	IEFSD41Q
IEFW1FAK	IEF41FAK
IEFW2FAK	IEFSD41Q
IEFW2FAK	IEF41FAK
IEFW21SD	IEFSD21Q
IEFW21SD	IEFSD557
IEFW22SD	IEFSD22Q
IEFW31SD	IEFSD31Q
IEFW32SD	IEFSD32Q
IEFW32SD	IEFSD33Q
IEFW41SD	IEFSD41Q
IEFW41SD	IEF41FAK
IEFW42SD	IEFSD42Q
IEFXA	IEFXCSSS
IEFXAMSG	IEFXAMSG
IEFXH000	IEFXH000
IEFXJMSG	IEFXJMSG
IEFXJX5A	IEFSD552
IEFXJX5A	IEFXJIMP
IEFXJ000	IEFXJFAK
IEFXJ000	IEFXJIMP
IEFXKMSG	IEFXKMSG
IEFXK000	IEFXKIMP
IEFXT000	IEFXT00D
IEFXT002	IEFXT002

(continued)

Entry Point or Control Section Name	Module Name
IEFXT003	IEFXT003
IEFXVMSG	IEFXVMSG
IEFXVNSL	IEFXVNSL
IEFXV001	IEFAVFAK
IEFXV001	IEFXV001
IEFXV002	IEFXV002
IEFXV003	IEFXV003
IEFX3000	IEFX300A
IEFX5000	IEFX5000
IEFYN	IEFYNIMP
IEFYNMSG	IEFYNMSG
IEFYP	IEFYPJB3
IEFYPMSG	IEFYPMSG
IEFYS	IEFYSVMS
IEFYT	IEFYTVMS
IEFZA	IEFZAJB3
IEFZG	IEFZGST1
IEFZGJ	IEFZGJB1
IEFZGMSG	IEFZGMSG
IEFZG2	IEFZGST2
IEFZH	IEFZHMSG
IEF085SD	IEFSD085
IEF086SD	IEFSD086
IEF589SP	IEF589SP
IEF850SD	IEFSD085
IEZDCODE	IEZDCODE
IEZNCODE	IEZNCODE
IGC0005B	IEFVSMBR
IGC00083	IGC0008C
IGC00090	IEFXQM00
IGC0103D	IGC0103D
IGC01090	IEFXQM01
IGC0203E	IEFWTP00
IGC02090	IEFXQM02
IGC0303E	IEFWTP01
IGC0403E	IEFWTP02
IGC1803D	IEESD571
IGC1903D	IEESD561
IGC3903D	IGC3903D
IGF08501	IGF08501
IGF08502	IGF08501
IGF13501	IGF13501
IGF2403D	IGF2403D
IGF2503D	IGF2503D
IGF2603D	IGF2603D
IGF29601	IGF29601
IGF29701	IGF29701
IGF55301	IGF55301
IKJNULL	IKJNULL
LOC	IEFLOCDQ
LOCCAN	IEFLOCDQ
LOCDQ	IEFLOCDQ
MSOFF	IEFXJMSG
MSRCV	IEFXJMSG
MSSYS	IEFXJMSG
SD304MG1	IEFSD312
SD304MG2	IEFSD312
SD305MG1	IEFSD312
SD55MSG1	IEFSD311
SD55MSG2	IEFSD311
SD55MSG3	IEFSD311
SMALLGO	IEFSD599
SMALTERM	IEFSD515
SMALTERM	IEFSD559
SPRINTER	IEFPRINT
STRMSG01	IEFYNMSG
VM7000	IEFVMLS1
VM7055	IEFVMLS1
VM7055AA	IEFVMLS1
VM7060	IEFVMLS1
VM7070	IEFVMLS1
VM7090	IEFVMLS1
VM7100	IEFVM2LS
VM7130	IEFVMLS1
VM7150	IEFVM3LS
VM7200	IEFVM4LS
VM7300	IEFVM5LS
VM7370	IEFVMLS1
VM7600	IEFVM76
VM7700	IEFVMLS1
VM7742	IEFJMIS1
VM7750	IEFVMLS1
VM7850	IEFVMLS1
VM7900	IEFVMLS1
VM7950	IEFVMLS1
XIIB32	IEFX5000
XPS631	IEFZHMSG
XTTEA0	IEFXT002
XTTEA1	IEFXT002
XTTEB3	IEFXT002
XTTP00	IEFXT00D
XTTRDJ	IEFXT002
XUUB00	IEFXT003
XUUHO6	IEFXT003
X33B42	IEFX300A
X55C86	IEFX5000
X55D3G	IEFX5000
YPPMSG1	IEFYPMSG
YPPMSG2	IEFYPMSG
ZGOE60	IEFZHMSG
ZGOK09	IEFZGST2
ZKOD1	IEFZHMSG
ZKOE1	IEFZHMSG
ZOOA1	IEFZGST2
ZOOE10	IEFZGST2
ZPOC10	IEFZGST2
ZPOQM	IEFZGJB1
ZPOQMGR1	IEFZGST1
ZPOQMGR2	IEFZGST2

(continued)

Module Descriptions

This section contains a brief description of each of the modules used by the MFT level of job management. Modules are listed alphamerically by module name; associated with each module is a descriptive name, which indicates the major component of the system to which the module belongs. Each description contains a brief statement of the purpose of the module. Where applicable, the description for each module includes the names of the module's entry points, the names of the modules to which the module passes control, the major tables and work areas to which the module refers, the module's attributes, and the names of the control sections that the module contains.

IEECIR50: Master Scheduler -- Resident WAIT Routine

This routine waits until a CSCB is chained representing a command to be processed, then passes control to IEECIR51 to process the command.

- Entry: IEECIR50
- Exits: IEECIR51, IEE00110.
- Attributes: Read-only, reentrant, resident
- Tables/Work Areas: CSCB, M/S resident data area
- Control Section: IEECIR50

IEECIR51: Master Scheduler -- Command Analyzer

This routine analyzes a command and passes control to the appropriate command processor or return control to IEECIR50.

- Entry: IEECIR51 from IEECIR50
- Exits: IEESD562, IEEDFIN1, or IEECIR50
- Attributes: Reentrant
- Tables/Work Areas: CSCB, M/S resident data area, UCM

IEEDFINA: Master Scheduler -- SMF MFT Storage Configuration Record Creation Routine

This routine creates the SMF dynamic storage configuration record for MFT. It is entered during SMF initialization and whenever a DEFINE Command is issued.

- Entry: IEEDFINA from IEESMFI2 during SMF initialization. From IEEDFIN9 whenever a DEFINE command is issued.
- Exit: Return to caller.
- Tables/Work Area: CVT, M/S resident data area, PIB, TCB, SMCA
- Attributes: Reentrant
- Control Sections: IEEDFINA

IEEDFINB: Master Scheduler -- System Reinitialization Routine 2

This routine waits for partitions to quiesce and then issues an ENQUEUE macro instruction specifying the boundary boxes.

- Entry: IEEDFINB
- Exits: IEEDFIN8
- Attributes: Read/only, reenterable
- Control Sections: IEEDFINB

IEEDFINC: Master Scheduler -- DEFINE Command Validity Check Routine (Core Storage)

This routine determines whether all information for the partition redefinition of core storage is correct.

- Entry: IEEDFINC
- Exits: IEEDFINB, IEEREXIT
- Tables/Work Areas: DFINDATA, CVT, M/S resident data area.
- Attributes: Read-only, reenterable
- Control Section: IEEDFINC

IEEDFIN1: Master Scheduler -- DEFINE Command Initialization Routine

This routine sets up data areas for partition definition, issues a DEFINE COMMAND BEING PROCESSED message to all active consoles, and passes control to the appropriate processing module.

- Entry: IEEDFIN1
- Exits: IEEDFIN3, IEEDFIN4, IEEDFIN5
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN1

IEEDFIN2: Master Scheduler -- DEFINE Command Syntax Check and Router Routine

This routine checks the syntax of DEFINE command statements. If a syntax error is discovered, the statement is ignored and an error message is issued. If the syntax is correct, the information is stored and

control is passed to the appropriate routine.

- Entry: IEEDFIN2, IEEDPART
- Exits: IEEDFIN5, IEEDFIN6, IEEDFIN7
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN2

IEEDFIN3: Master Scheduler -- DEFINE Command Validity Check Routine (Processor Storage)

This routine determines whether all information for the partition redefinition of processor storage is correct.

- Entry: IEEDFIN3
- Exits: IEEDFINC, IEEREXIT
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN3

IEEDFIN4: Master Scheduler -- DEFINE Command Listing Routine

This routine lists partition definitions.

- Entry: IEEDFIN4
- Exits: IEEDFIN3, IEEDFIN5
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN4

IEEDFIN5: Master Scheduler -- DEFINE Command Message Routine

This routine contains texts for operator messages required for DEFINE command processing. The message is constructed according to a code passed by the calling routine. IEEDFIN5 issues the requested message and passes control to IEEDFIN2 or the dispatcher.

- Entry: IEEDFIN5
- Exits: IEEDFIN1, IEEDFIN2 or return to calling program
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN5

IEEDFIN6: Master Scheduler -- Time-Slice Syntax Check Routine

This routine checks the TMSL subparameters for proper syntax.

- Entry: IEEDFIN6
- Exits: IEEDFIN2, IEEDFIN5, IEEDPART
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN6

IEEDFIN7: Master Scheduler -- Keyword Scan Routine

This routine checks keyword parameters for syntax errors. If a syntax error is discovered, the erroneous entry and all following entries are ignored, and an error message is generated. If the syntax is correct, the information is stored.

- Entry: IEEDFIN7
- Exits: IEEDFIN2, IEEDFIN3, IEEDFIN4, IEEDFIN5, IEEDPART
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN7

IEEDFIN8: Master Scheduler -- System Reinitialization Routine - Part 1

This routine assigns protection keys, marks dispatchable partitions that are not of zero size, and checks that the number of problem program partitions does not exceed 15. If no error is found, IEEDFIN8 builds request blocks and boundary boxes and updates the PIB and the TCBPIB field for the defined partition.

- Entry: IEEDFIN8
- Exits: IEEDFIN9, IEEREXIT
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN8

IEEDFIN9: Master Scheduler -- Command Final Processor Routine

This routine updates the time-slice control element, if time-slicing is specified, and issues a message to all active consoles that processing is complete.

- Entry: IEEDFIN9, IEEREXIT
- Exits: IEEDFIN5, IEEDFINA
- Attributes: Read-only, reenterable
- Control Section: IEEDFIN9

IEELOG02: Master Scheduler -- Log Open Initialization Module

This routine opens the system log at IPL time.

- Entry: IEELOG02

- Exit: IEFSD569
- Tables/Work Areas: CVT, UCB, UCM, TIOT, M/S resident data area, JFCB, IEELCA, DCB.
- Attributes: Refreshable
- Control Section: IEELOG02

IEELWAIT: Master Scheduler -- Log Wait and Writer Module

This module writes data from the log buffer to the system log.

- Entry: IEELWAIT
- Exit: To Dispatcher
- Tables/Work Areas: CVT, LCA, MRC
- Attributes: Resident
- Control Section: IEELWAIT

IEEPSN: System Task Control -- Interpreter Post Scan Exit Routine

This routine verifies that the program name in the EXEC statement of a starting procedure is in the table of system tasks that qualify to be started.

- Entry: IEEPSN
- Exit: Return to caller
- Attributes: Reenterable
- Tables/Work Areas: Linkage table
- Control Section: IEEPSN

IEESD561: SVC 34 -- STOP INIT and START Command Processor (Part 1)

This routine initially processes the STOP INIT and START commands.

- Entry: IGC1903D
- Exit: IEE3903D, IEE0503D for error messages
- Tables/Work Areas: CVT, XSA
- Attributes: Read-only, transient
- Control Section: IGC1903D

IEESD562: Master Scheduler -- Syntax Check Routine

This routine checks syntax of the command and sets internal codes for queue search, if required.

- Entry: IEESD562
- Exits: XCTL to IEESD563 for queue search, to IEESD566 for DISPLAY or MONITOR active, to IEEXENDA for DISPLAY CONSOLES, to IEESD565 for syntax error
- Attributes: Read-only, reenterable
- External References: None
- Control Section: IEESD562

IEESD563: Master Scheduler -- Queue Search Setup Routine

This routine determines which queue is to be searched, reads and scans the queue control record, establishes parameters for the search, and transfers control to the queue search module. IEESD563 will write out updated queue control records.

- Entry: IEESD563
- Exits: XCTL to IEESD564 to search queue; XCTL to IEESD565 at completion, XCTL to IEESD575 for CANCEL; XCTL to IEESD583 or IEESD584 for D Q or D N
- Tables/Work Areas: QCR, QMPA, CVT, CSCB
- Attributes: Read-only, reenterable
- Control Section: IEESD563

IEESD564: Master Scheduler -- Queue Search Module

This routine searches the work queues for the execution of the queue manipulation commands.

- Entry: IEESD564
- Exit: XCTL to IEESD563
- Tables/Work Areas: QCR, CSCB, CVT, QMPA, XSA
- Attributes: Read-only, reenterable
- Control Section: IEESD564

IEESD565: Master Scheduler -- Service Routine

This routine frees storage obtained by IEESD563, links to the queue manager to enqueue an entry or queue control record on SYS1.SYSJOBQE, or links to write a message.

- Entry: IEESD565
- Exit: Return to caller

- Tables/Work Areas: QMPA, CSCB, QCR, CVT
- Attributes: Read-only, reenterable
- External References: IEFQMNQ2, IEE0503D
- Control Section: IEESD565

IEESD566: Master Scheduler -- DISPLAY A Routine

This routine builds a table and constructs operator messages according to the processing required by a DISPLAY A command.

- Entry: IEESD566
- Exit: Return to caller (IEECIR50)
- Tables/Work Areas: QMPA, CSCB, XSA, QCR, CVT
- Attributes: Read-only, reenterable
- Control Section: IEESD566

IEESD568: Nucleus -- Master Scheduler Resident Data Area

This routine contains the master scheduler resident data area.

- Entry: IEEMSER
- Exit: None
- Attributes: Not reusable
- Control Section: IEEMSER

IEESD571: SVC 34 -- DEFINE, MOUNT Routine

This routine schedules the execution of the DEFINE and MOUNT commands.

- Entry: IGC1803D
- Exits:
  MOUNT - XCTL to IGC0103D
  DEFINE - Return to caller
  XCTL to IEE0503D and IEE2103D due to error.
- Tables/Work Areas: PIB, M/S resident data area, CVT
- Attributes: Reenterable
- Control Section: IGC1803D

IEESD575: Master Scheduler -- Queue Scratch Setup Routine

This routine sets up the scratch parameter list.

- Entry: IEESD575 from IEESD563 or from IEESD583
- Exit: XCTL to IEESD581 to Scratch or to IEESD576 to delete queue entries.
- Tables/Work Areas: CSCB, CVT, JCT, JFCB, QMPA, SCT, SIOT, TIOT, UCB, UCB Look up Figure
- Attributes: Read-only, reenterable
- Control Section: IEESD575

IEESD576: Master Scheduler -- Queue Alter Delete Routine

This routine goes to the Queue Manager delete routine IEFQDELQ to delete queue entries.

- Entry: IEESD576 from IEESD575
- Exit: XCTL to IEESD578 for message class setup.
- Tables/Work Areas: CSCB, CVT, DSB, JCT, SCD, SMB
- Attributes: Read-only, reenterable
- Control Section: IEESD576

IEESD577: Master Scheduler -- Queue Restart Enqueue Routine

This routine links to the queue manager enqueue routine IEFQMNQQ to enqueue data sets for a canceled restarting job.

- Entry: IEESD577 from IEESD578
- Exit: XCTL to IEESD579
- Tables/Work Areas: CSCB, QMPA, SCD, SMB
- Attributes: Read-only, reenterable
- Control Section: IEESD577

IEESD578: Master Scheduler -- Queue Message Class Setup

This routine zeroes out the DSB's in the message class and sets up the QMPA for enqueuing the message class.

- Entry: IEESD578 from IEESD576
- Exit: XCTL to IEESD579 to enqueue the message class; XCTL to IEESD577 to enqueue the sysout data sets for a restarting job; XCTL to IEESD580 for message, XCTL to IEESD563 for queue search.

- Tables/Work Areas: CSCB, CVT, JCT, QMPA, SCD, SMB
- Attributes: Read-only, reenterable
- Control Section: IEESD578

IEESD579: Master Scheduler -- Queue SMB Routine

This routine places the appropriate CANCEL message in the SMB and goes to the queue manager enqueue routine to enqueue the message class. The operator message is also issued from this routine.

- Entry: IEESD579 from IEESD577 or IEESD578
- Exit: Return to IEECIR50
- Tables/Work Areas: CSCB, CVT, QMPA, SMB
- Control Sections: IEESD579, IEEMSWTO

IEESD580: Master Scheduler -- Message Routine

This routine issues the CANCEL WTO for those jobs that do not have the message class.

- Entry: IEESD580 from IEESD578
- Exit: Return to IEECIR50
- Tables/Work Areas: CSCB, QMPA
- Control Sections: IEESD580

IEESD581: Master Scheduler -- Queue Scratch Routine

This routine issues the SCRATCH macro (SVC 29) or an error message.

- Entry: IEESD581 from IEESD575
- Exit: XCTL to IEESD575
- Attributes: Read-only, reenterable
- Control Section: IEESD581

IEESD582: Master Scheduler -- ECB/IOB Construction Routine

This routine builds an event control block and an input/output block used in a queue search.

- Entry: IEESD582 or IEESD82A from IEESD562
- Exit: XCTL to IEESD563 for a queue search or to IEESD565 for a message
- Tables/Work Areas: CSCB, XSA
- Attributes: Reentrant
- Control Sections: IEESD582

IEESD583: Master Scheduler -- Queue Search Return Routine

This routine analyzes the results of a queue search made by the queue search routine, IEESD564.

- Entry: IEESD583
- Exits: XCTL to IEESD563 for a queue search; XCTL to IEESD565 for a message; XCTL to IEESD575 to cancel processing.
- Tables/Work Areas: CSCB, CVT, QCR, QMPA
- Attributes: Reentrant
- Control Section: IEESD583

IEESD584: Master Scheduler -- D Q/D N Message Setup Routine

This routine writes the control and label lines for the DISPLAY Queue and the DISPLAY jobname commands.

- Entry: IEESD584 from IEESD563
- Exit: XCTL to IEESD564 or IEESD583
- Tables/Work Areas: CSCB
- Attributes: Reentrant
- Control Section: IEESD584

IEESD590: System Task Control -- Write TIOT on Disk

This routine writes the TIOT which is used by Job Selection (IEESD510) and checks for a small partition writer.

- Entry: IEESD590
- Exits: Return to IEFSD589 or XCTL to IEESD591
- Tables/Work Areas: TIOT, SPIL
- Attributes: Reenterable
- Control Section: IEESD590

IEESD591: System Task Control -- Linker Routine

This routine transfers control between system task control and a system task or a transient reader.

- Entry: IEESD591, IEE591SD
- Exit: Return to IEFSD589 or IEF589SP
- Tables/Work Areas: CSCB, CVT, PIB, IWA, QMPA
- Attributes: Reenterable
- Control Section: IEESD591

IEESD592: System Task Control -- POST Routine

This routine posts the "no work" ECB in the PIB and clears the partition for further processing.

- Entry: IEESD592
- Exit: Return to IEFSD587 or IEFSD589
- Tables/Work Areas: None
- Attributes: Reenterable
- Control Section: IEESD592

IEESMFAL: SVC 83 -- SMF Allocation Routine

This routine allocates devices for the SMF data sets.

- Entry: IEESMFAL
- Exit: IEESMFOP
- Attributes: Reentrant
- Tables/Work Areas: SMCA, CVT
- Control Sections: IEESMFAL

IEESMFIT: Master Scheduler -- SMF Initialization Routine (1)

This routine obtains storage for and initializes the SMCA, and reads the SMFDEFLT parameters from SYS1.PARMLIB.

- Entry: IEESMFIT, IEESMFI4
- Exit: IEFSD569, IEESMFOI, IEESMFI2, IEESMFI3
- Attributes: Non-reentrant
- Tables/Work Areas: CVT, DCB, JFCB, M/S Resident Data Area, SMCA, TIOT, UCB
- Control Sections: IEESMFIT

IEESMFI2: Master Scheduler -- SMF Initialization Routine (2)

This routine constructs the SMF IPL and initial online I/O device records, and enqueues the 10-minute TQE on the timer queue.

- Entry: IEESMFI2
- Exit: To IEEDFINA, IEESMFIT
- Attributes: Non-reentrant
- Control Sections: IEESMFI2

IEESMFI3: Master Scheduler -- SMF Parameter Processing Routine

This routine processes the SMFDEFLT parameters and/or the SMF parameters entered from the operator's console.

- Entry: IEESMFI3, IEESMFIO, IEESMFMS
- Exit: IEESMFIT (at entry point IEESMFI4)
- Attributes: Serially Reusable
- Tables/Work Areas: CVT, M/S Resident Data Area, SMCA, UCB
- Control Section: IEESMFI3

IEESMFOI: Master Scheduler -- SMF Open Initializer

This routine stores the DCBs and JFCBs for the SMF data sets.

- Entry: IEESMFOI
- Exit: IEESMFIT
- Attributes: Serially reusable
- Tables/Work Areas: DCB, JFCB, SMCA, TIOT
- Control Sections: IEESMFOI

IEESMFOP: SVC 83 -- Open Routine

This routine opens the SMF data sets, and switches between the primary and alternate data sets.

- Entry: IEESMFOP
- Exit: IEESMFAL, return to caller
- Tables/Work Areas: DCB, JFCB, SMCA, TIOT
- Attributes: Reentrant
- Control Sections: IEESMFOP

IEESMFWT: SMF Writer Routine

This routine writes the contents of the SMF buffer in the SMF data set.

- Entry: IEESMFWT
- Exit: return to caller
- Tables/Work Areas: CVT, DCB, SMCA
- Attributes: Reentrant
- Control Section: IEESMFWT

IEEVACTL: System Task Control -- Allocation Interface Routine

This routine sets up the interface between system task control and the I/O device allocation routine.

- Entry: IEEVACTL
- Exits: To IEFSD21Q, IEEVSMSG, IEEVTCTL, or IEEVSMBA
- Attributes: Reenterable
- Control Section: IEEVACTL

IEEVICLR: Internal JCL Reader

This routine reads the internal job control language used in starting a reader or writer.

- Entry: IEEVICLR
- Exit: Return to caller
- Tables/Work Areas: DCBD
- Attributes: Read-only, reenterable
- Control Section: IEEVICLR

IEEVJCL: System Task Control -- JCL Edit Routine

This routine constructs the internal job control language used in the START reader and START writer command execution routines.

- Entry: IEEVJCL, from IEEVSTAR
- Exit: XCTL to IEEVRCTL
- Tables/Work Areas: SDT, CSCB, JSEL, JSOL, JSXL
- Attributes: Reenterable
- Control Section: IEEVJCL

IEEVLDSP: Master Scheduler -- Log Dispatcher Routine

This routine puts the log data set on the system output queue.

- Entry: IEEVLDSP
- Exit: Master Scheduler
- Tables/Work Areas: M/S Resident Data Area, CT, IEELCA, UCB, JFCB.
- Attributes: Reentrant
- Control Section: IEEVLDSP

IEEVLIN: Master Scheduler -- Log Initialization Routine

This routine initializes the system log.

- Entry: IEEVLIN
- Exit: IEFSD569, IEEVLIN2
- Tables/Work Areas: UCM, CVT, UCB, TIOT, M/S resident data area, IEELCA.
- Attributes: Refreshable
- Control Section: IEEVLIN

IEEVLNKT: System Task Control -- Link-Figure Module

This module consists of a table of the program names that are specified in the EXEC statement for a system task.

- Entry: IEEVLNKT
- Attributes: Non-executable
- Control Section: IEEVLNKT

IEEVLOUT: Log Data Set Reinitialization Routine

This routine opens and closes the log data set to reinitialize the DS1LSTAR and DS1TRBAL fields of the DSCB associated with the log data set.

- Entry: IEEVLOUT, from IEFSD171
- Exit: IEFSD171
- Tables/Work Areas: CVT, DSCB, LCA, M/S Resident Data Area
- Attributes: Reenterable
- Control Section: IEEVLOUT

IEEVOMSG: System Task Control -- Message Writing Routine

This routine assembles and writes messages to the operator.

- Entry: IEEVOMSG

- Exit: Return to caller
- Control Section: IEEVOMSG

IEEVPRES: Master Scheduler -- Public/Private Interface Routine

This routine sets up the interface for volume attribute setting routine IEFPRES.

- Entry: IEEVPRES
- Exit: To IEFPRES
- Tables/Work Areas: UCB
- Attributes: Reentrant
- Control Section: IEEVPRES

IEEVRC: System Task Control -- Interpreter Exit Routine

This routine performs general cleanup functions following the processing of the reader/interpreter.

- Entry: IEEVRC
- Exit: XCTL to IEEVACTL, Return to IEFSD589
- Tables/Work Areas: CSCB, CVT, IEEPSN work area, JSEL, JSXL, M/S Resident Data Area, NEL, QMPA, JSWA
- Attributes: Reenterable
- Control Section: IEEVRC

IEEVRCTL: System Task Control -- Reader/Interpreter Control Routine

This routine provides an interface between system task control and an interpreter.

- Entry: IEEVRCTL
- Exits: XCTL to IEFVH1
- Tables/Work Areas: CVT, CSCB, JSEL, JSOL, JSXL, JSWA, TIOT, OPT
- Control Section: IEEVRCTL

IEEVRFRX: Master Scheduler -- Table Lookup Routine

This routine can be used to obtain the following information; the CVT address, the contents of a CVT entry, or the contents at the CVT pointer address, a pointer to the TCB or the RB, the TIOT pointer, the TIOT entry, the TIOT TTR, or the TIOT UCB pointer. The routine can also be used to insert a TIOT pointer, a TIOT TTR, or a TIOT UCB pointer in the CVT.

- Entry: IEEVRFRX
- Exit: Return to calling program
- Tables/Work Areas: CVT, TCB, RB, TIOT, UCB
- Attributes: Reenterable
- Control Section: IEEVRFRX

IEEVRJCL: System Task Control -- Internal JCL Reader

This routine reads the internal job control language used in starting a reader or writer.

- Entry: IEEVRJCL
- Exit: Return to caller
- Tables/Work Areas: DCBD
- Attributes: Read-only, reenterable
- Control Section: IEEVRJCL

IEEVSMBA: System Task Control -- QMPA Builder

This routine constructs a queue manager parameter area (QMPA) referring to the message class queue for the use of the I/O Device Allocation routine.

- Entry: IEEVSMBA
- Exit: To IEEVACTL
- Tables/Work Areas: QMPA, LCT, SMB, IOB
- Control Section: IEEVSMBA

IEEVSMSG: System Task Control -- Message Writer Routine

This routine writes messages to the operator as required by the master scheduling task and system task control.

- Entry: IEEVSMSG
- Exit: Return to caller
- Control Section: IEEVSMSG

IEEVSTAR: System Task Control -- START Command Syntax Check Routine

This routine checks the syntax of a START command, and builds a start descriptor table (SDT) containing the parameters of the command.

- Entry: IEEVSTAR

- Exits: To IEEVJCL, or IEE0503D
- Tables/Work Areas: SDT, M/S Resident Data Area, CVT, UCB XSA, and CSCB.
- Attributes: Reenterable
- Control Section: IEEVSTAR

IEEVTCTL: System Task Control -- Termination Interface Routine

This routine initializes the necessary tables for terminating a task that was established via a START command, and releases storage obtained by IEEVSTAR for the task's JSCB and WTPCB.

- Entry: IEEVTCTL
- Exit: IEFQMSS, IEEVSMSG, IEFW42SD, IEFQDELE, IEEPRTN2, IEEVOMSG
- Tables/Work Areas: TCB, JCT, SCT, LCT, and CSCB
- Attributes: Reenterable. Character Dependence Type C
- Control Section: IEEVTCTL

IEEXEDNA: Master Scheduler -- DISPLAY CONSOLES Processor

This routine processes the DISPLAY command with the CONSOLES operand and displays the system console configuration on the requesting console.

- Entry: IEEXEDNA from IEESD562
- Exit: To IEECIR50
- Attributes: Reentrant
- Control Sections: 1EEXEDNA

IEE00110: SVC 110 -- M/S Router

This routine passes control to the display processors for DISPLAY, DUMP, and MONITOR commands that involve status displays.

- Entry: IEE00110 (this routine is the first load of SVC110; SVC 110 is issued by IEECIR50.)
- Exits: IEE10110, IIE20110, IEE40110, IEE60110
- Attributes: Reentrant, transient
- Tables/Work Areas: CSCB, M/S Resident Data Area, XSA

IEE0303D: SVC 34 -- CSCB and CIB Chain Manipulator

This routine manipulates the CSCB and CIB chain as requested by the caller of SVC 34.

- Entry: IEE0303D
- Exit: To IEE5403D, or return to caller
- Tables/Work Areas: CVT, M/S resident data area, CSCB, XSA
- Control Section: IEE0303D

IEE0303F: SVC 36 -- WRITE-TO-LOG

This module copies text records from an input area to the log buffer and posts the log ECB when the buffer is full.

- Entry: IEE0303F
- Exit: Returns to Master Scheduler, IEE0403F.
- Tables/Work Areas: IEEBASEA, IEELCA, CVT
- Control Section: IEE0303F

IEE0403D: SVC 34 -- Router Routine

This routine identifies the command verb, ensures that the console has authority to enter the command, and passes control to the appropriate routine.

- Entry: IEE0403D
- Exit: Depending on command verb, via XCTL to another SVC 34 module
- Tables/Work Areas: M/S resident data area, XSA, CSCB
- Control Section: IEE0403D

IEE0403F: SVC 36 (Load 2) -- Log Buffer Management Module

This module opens, closes, and switches system log buffers.

- Entry: IEE0403F
- Exit: IEE0303F
- Tables/Work Areas: IEEBASEA, IEELCA, UCB, JFCB, DCB, CVT, TIOT.
- Attributes: Reentrant
- Control Section: IEE0403F

IEE0503D: SVC 34 -- Message Assembly Routine

This routine assembles and edits messages for the command scheduling routine, and writes the messages to the operator.

- Entry: IEE0503D
- Exit: Branch on register 14
- Attributes: Reenterable, read-only
- Control Section: IEE0503D

IEE0603D: SVC 34 -- SET Command Routine, Part I

This routine processes the SET command operands, CLOCK and DATE.

- Entry: IEE0603D
- Exits: To IEE0503D or IEE8603D
- Tables/Work Areas: XSA, CVT, M/S resident data area
- Attributes: Reenterable, self-relocating, read only transient
- Control Section: IEE0603D

IEE0703D: SVC 34 -- STOP and MODIFY Scheduling

This routine schedules the execution of the STOP and MODIFY commands by finding and updating the appropriate CSCB and by issuing a POST macro instruction to the master scheduling task.

- Entry: IEE0703D
- Exits: Branch on register 14, or XCTL to IEE0503D or IEE2103D.
- Tables/Work Areas: M/S Resident Data Area, XSA, CVT, CSCB
- Attributes: Reenterable, self-relocating, read-only, transient
- Control Section: IEE0703D

IEE0803D: SVC 34 -- CSCB Creation Routine

This routine schedules the execution of commands that cannot be completely processed by the command scheduling routines. It performs this function by adding a CSCB to the CSCB chain and issuing a POST macro instruction to the master scheduling task.

- Entry: IEE0803D
- Exit: IEE0503D, IEE2103D, or return to caller
- Tables/Work Areas: XSA, M/S resident data area, CVT, CSCB, and UCM
- Attributes: Reenterable, transient, partially disabled.
- Control Section: IEE0803D

IEE0903D: SVC 34 -- Timer Maintenance Routine

This routine processes the date and time operands of the SET command.

- Entry: IEAQOT00
- Exit: SVC 3
- Tables/Work Areas: CVT
- Attributes: Reenterable, supervisor state, disabled for system interrupts, transient
- Control Section: IEAQOT00

IEE10110: SVC 110 -- DISPLAY C,K Routine (Load 1)

This routine begins processing of the DISPLAY C,K command by building the title and label lines of the display and writing them to the operator's console by means of the WTO macro instruction.

- Entry: IEE10110 from IEE00110
- Exit: IEE11110
- Attributes: Reentrant, privileged, transient
- Tables/Work Areas: CSCB, XSA

IEE1103D: SVC 34 -- VARY and UNLOAD Scan Routine for Non-MCS Systems

This routine examines the command and its operand.

- Entry: IEE1103D
- Exit: IEE2203D for multiprocessing VARY commands, IEE2303D for VARY ONLINE and CONSOLES operands when SMF is present, to IEE3103D for all other VARY operands and UNLOAD, and to IEE0503D for errors.
- Tables/Work Areas: XSA, CVT, M/S resident data area, and UCM.
- Attributes: Reenterable, self-relocating, read-only, and transient.

IEE11110: SVC 110 -- DISPLAY C,K Routine (Load 2)

This routine continues processing of the DISPLAY C,K command by building the middle portion of the display and writing the lines to the operator's console by means of the WTO macro instruction.

- Entry: IEE11110 from IEE10110
- Exits: IEE12110
- Attributes: Reentrant, privileged, transient
- Tables/Work Areas: CSCB, XSA
- Control Sections: IEE1103D

IEE1203D: SVC 34 -- REPLY Processor

This routine checks the validity of the operator's REPLY command, and moves the operator's REPLY (if valid) to the buffer of the user that issued the respective WTOR.

- Entry: IEE1203D
- Exit: To IEE1B03D to issue error message
- Tables/Work Areas: CVT, UCM, WQE, RQE, CXSA

IEE12110: SVC 110 -- DISPLAY C,K Routine (Load 3)

This routine completes building of the DISPLAY C,K status display.

- Entry: IEE12110
- Exits: Return to caller (IEE00110)
- Attributes: Reentrant, privileged, transient
- Tables/Work Areas: CSCB, XSA
- Control Section: IEE1203D

IEE1403D: SVC 34 -- HALT Routine

This routine processes the HALT and SWITCH commands. The SWITCH command will result in special processing for SMF if it is in the system. The HALT command will cause an SVC 76 to be issued for a statistics update, the LOG data set to be closed, as well as any SMF processing that may be required.

- Entry: IEE1403D from IEE0403D
- Exit: IEE1303D, TCAM on IEE0503D
- Tables/Work Areas: CVT, M/S Resident Data Area, SMCA, UCB
- Attributes: Reentrant
- Control Section: IEE1403D

IEE1603D: SVC 34 -- LOG and WRITELOG Processor Routine

This routine issues a WTL macro instruction when a LOG command is issued, and stores the WRITELOG command and posts the Log ECB, for WRITELOG processing.

- Entry: IEE1603D, from IEE0403D
- Exit: IEE0503D for errors, and return to caller of SVC 34.
- Tables/Work Areas: XSA, CVT, LCA, and M/S resident data area.
- Attributes: Reentrant, self-relocating, read-only, and transient.
- Control Sections: IEE1603D

IEE1703D: SVC 34 -- VARY ONGFX/OFFGFX

This routine processes the GVARY command. It checks the parameters for validity and if an error is found, it passes control to IEE0503D via an XCTL macro instruction. If the parameters are valid, the routine sets appropriate bits in the Overall Control Table (OCT) of the GFX reader task. It then issues a POST macro instruction on the ECB in the OCT for each graphics device (2250) placed in the online status.

- Entry: IEE1703D
- Exit: IEE0503D, return to issuer of SVC 34
- Tables/Work Areas: CVT, OCT, XSA
- Attributes: Reenterable, read-only, self-relocating
- Control Section: IEE1703D

IEE1A03D: SVC 34 -- MCS REPLY Processor Routine

The purpose of this routine is to process valid operator replies to WTOR macro instructions in an MCS environment.

- Entry: IEE1A03D
- Exit: To IEE1B03D
- Tables/Work Areas: CVT, UCM, WQE, RQE, XSA

- Attributes: Reenterable, transient
- Control Sections: IEE1A03D

IEE1B03D: SVC 34 -- MCS Reply Message Routine

This routine assembles, edits, and broadcasts the accepted reply to a WTOR macro instruction for the MCS Reply Processor routine (module IEE1A03D) of the Command Scheduling routine, and to write error messages to the operator whose command is in error.

- Entry: IEE1B03D, from IEE1A03D
- Exit: Return to the caller of SVC 34
- Tables/Work Areas: CVT, UCM, RQE, WQE, XSA
- Attributes: Reenterable, transient
- Control Sections: IEE1B03D

IEE20110: SVC 110 -- DISPLAY U Routine (Load 1)

This routine begins processing of the DISPLAY U command by checking the command syntax and the UCB for the first device to be included in the status display.

- Entry: IEE20110 from IEE00110 (SVC 110 issued by IEECIR51
- Exits: IEE21110, IEE22110, IEE23110
- Attributes: Reentrant, privileged, transient
- Tables/Work Areas: CSCB, UCB, Workarea obtained by GETMAIN

IEE2103D: SVC 34 -- Message Assembly Routine

This routine assembles and edits messages for the command scheduling routine, and writes the messages to the operator.

- Entry: IEE2103D
- Exit: Branch on register 14
- Attributes: Reenterable, self-relocatory, read-only, transient
- Control Section: IEE2103D

IEE21110: SVC 110 -- DISPLAY U Routine (Load 2)

This routine continues processing of the DISPLAY U command by building display lines using information obtained from the UCB.

- Entry: IEE21110
- Exit: IEE23110, IEE22110
- Attributes: Reentrant, privileged, transient
- Tables/Work Areas: CSCB, Workarea obtained by GETMAIN, UCB, Device Name Table

IEE22110: SVC 110 -- DISPLAY U Routine (Load 3)

This routine issues information pertaining to the DISPLAY U commands. It also builds line entries for data cell entries in the display and performs final cleanup (freeing the workarea) before returning to IEE00110.

- Entry: IEE22110 from IEE20110 or IEE23110 or IEE2110
- Exit: IEE23110, IEE2110, or return to IEE00110 via XCTL
- Attributes: Reentrant, privileged, type 4 SVC
- Tables/Work Areas: CSCB, UCB Workarea from GETMAIN

IEE2303D: SVC 34 -- SMF VARY Processor

This routine initially processes the ONLINE and CONSOLES operand of the VARY command when the system has the SMF option. It builds and issues an SMF record for each device placed in online status.

- Entry: IEE2303D from IEE1103D or IEE3303D
- Exit: IEE3103D, IEE4203D, or IEE4403D
- Tables/Work Areas: CVT, SMCA, XSA
- Attributes: Reentrant, read-only, self-relocating
- Control Sections: IEE2303D

This routine continues processing of the

IEE23110: SVC 110 -- Display U Routine (Load 4)

This routine continues processing of the DISPLAY U command by locating UCBs that satisfy the command and by writing lines of the display (create by loads 2 and 3) to the operator's console.

- Entry: IEE23110 from IEE20110, IEE21110 and IEE22110
- Exits: IEE21110 and IEE22110

- Attributes: Reentrant, privileged, transient
- Tables/Work Areas: UCB, CSCB, Workarea from GETMAIN

IEE2803D: SVC 34 -- CANCEL Command Processor

This routine checks the syntax of and processes the CANCEL command.

- Entry: IEE2803D
- Exit: BALR to ABTERM
  XCTL to IEE0803D
  XCTL to IEE0503D and IEE2103D for messages
- Tables/Work Areas: CSCB, TCB, M/S Resident data area
- Attributes: Read-only, reenterable
- Control Section: IEE2803D

IEE2903D: SVC 34 -- Display Requests Routine

This routine displays to the requesting operator the ID of all outstanding WTORs, the unit name of each device for outstanding MOUNT messages, and an indication as to whether any AVR mount messages are pending.

- Entry: IEE2903D, from IEE0803D
- Exit: Return to caller of SVC 34
- Tables/Work Areas: Message work area
- Attributes: Reentrant, Refreshable, transient
- Control Sections: IEE2903D

IEE3103D: SVC 34 -- VARY and UNLOAD Processor Routine

This routine processes the UNLOAD command, all VARY command operands in a system without the MCS option, and VARY ONLINE and OFFLINE operands for non-console devices in a system with the MCS option.

- Entry: IEE3103D, initially from IEE1103D or IEE2303D and returns from IEE4603D.
- Exit: Return to Caller
- Tables/Work Areas: XSA, UCM, CVT, M/S resident data area, and UCB.
- Attributes: Reentrant, self-relocating, read-only, transient.
- Control Sections: IEE3103D

IEE3203D: SVC 34 - VARY Keyword Router Routine

This routine identifies the first keyword in a VARY command, checks its validity, and passes control to the appropriate command keyword processing routine.

- Entry: IEE3203D from IEE0403D
- Exit: To IEE1103D, IEE3303D, IEE4303D, IEE4703D, IEE2403D, IEE1703D
- Tables/Work Areas: XSA
- Attributes: Reentrant, read-only, self-relocating
- Control Sections: IEE3203D

IEE3303D: SVC 34 -- MCS VARY Syntax Check Routine

This routine scans the command syntax in an MCS environment.

- Entry: IEE3303D from IEE3203D
- Exit: IEE2303D, IEE2203D, IEE4203D, or IEE4403D
- Tables/Work Areas: XSA, UCB, UCM, CVT
- Attributes: Reenterable, self-relocating, read only
- Control Sections: IEE3303D

IEE3503D: SVC 34 -- DISPLAY Command Router Routine

This routine examines the operand of DISPLAY commands and passes control to the routine that processes the command. It also processes all MONITOR commands.

- Entry: IEE3503D from IEE0403D
- Exits: IEE7503D, IEE1303D, IEE5803D, IEE2903D, IEE0803D, IEE0503D, IEE8503D, IEE2103D or return to caller
- Tables/Work Areas: M/S resident data area, XSA, CVT, and UCM
- Attributes: Reenterable, transient
- Control Section: IEE3503D

IEE3903D: SVC 34 -- STOP INIT and START Command Processing Routine (Part 2)

This routine completes the processing of the STOP INIT and START commands.

- Entry: IGC3903D
- Exit: Return to caller
- Tables/Work Areas: CVT, XSA, CSCB, PIB, M/S resident data area
- Attributes: Read-only, transient
- Control Section: IEE3903D

IEE40110: SVC 110 -- DISPLAY PFK Routine

This routine builds a display of PFK definitions, requested by the DISPLAY PFK command, and writes the display to the operator's console.

- Entry: IEE40110
- Exit: IEE00110, for return to supervisor
- Attributes: Reentrant, transient
- Tables/Work Areas: DCM

IEE4103D: SVC 34 -- Hardcopy Message Issuing Routine

This routine issues messages concerning the status of the hard copy log.

- Entry: IEE4103D, from IEE4703D
- Exit: Return to caller
- Tables/Work Areas: XSA, message area, UCB, CVT, XSA, and UCM
- Attributes: Reentrant, transient
- Control Sections: IEE4103D

IEE4203D: SVC 34 -- VARY Secondary Scanner Routine

This module performs authority and operand validity checking, and passes control to the routine that will process the command.

- Entry: IEE4203D, from IEE3303D or IEE2303D
- Exit: To IEE3103D, IEE4603D, or IEE4903D
- Tables/Work Areas: XSA, CVT, UCM, and UCB.
- Attributes: Reenterable, self-relocating, read-only, transient
- Control Sections: IEE4203D

IEE4303D: SVC 34 -- VARY MSTCONS Routine

This routine processes the VARY MSTCONS command.

- Entry: IEE4303D from IEE3203D VARY CONSOLE
- Exit: To IEE0503D or IEE2103D on errors, SVC 72 to Console Switch Routine (module IEECMCSW) and upon return to caller of SVC 34
- Tables/Work Areas: UCB, CVT, XSA, and UCM
- Attributes: Reentrant, self-relocating, read-only, transient
- Control Sections: IEE4303D

IEE4403D: SVC 34 -- VARY Scan Routine

This routine determines the validity of VARY CONSOLE and sets appropriate bits in the XSA.

- Entry: IEE4403D, from IEE3303D or IEE2303D
- Exits: IEE4203D, IEE0503D, IEE2103D
- Tables/Work Areas: XSA, UCM, CVT, and UCB
- Attributes: Reentrant, transient
- Control Sections: IEE4403D

IEE4503D: SVC 34 -- Periodic STOP/STOPMN Command Handler Routine

This routine syntax scans the STOP and MODIFY commands and processes the STOP JOBNAMES/STATUS/SPACE/DSNAME commands in a non-MCS environment.

- Entry: IEE4503D, from IEE0403D
- Exits: IEE0703D, IEE0503D, IEE2103D, IEE5503D, IEE7503D, return to caller
- Tables/Work Areas: XSA, M/S resident data area, CVT, and UCM
- Attributes: Reentrant, self-relocating, read-only, transient
- Control Section: IEE4503D

IEE4603D: SVC 34 -- VARY ONLINE/OFFLINE Routing Routine for Console Devices

This routine processes VARY ONLINE/OFFLINE for all MCS consoles.

- Entry: IEE4603D, from IEE4203D to process VARY ONLINE-OFFLINE

- Exit: IEE3103D or return to caller
- Tables/Work Areas: XSA, CVT, UCB, UCM, and M/S resident data area
- Attributes: Reentrant, self-relocating, read-only, transient
- Control Sections: IEE4603D

IEE4703D: SVC 34 -- VARY HARDCPY Processor Routine

This routine processes VARY HARDCPY commands.

- Entry: IEE4703D from IEE3203D
- Exit: To IEE5703D or IEE7203D
- Tables/Work Areas: XSA, UCM, M/S resident data area, CVT and UCB
- Attributes: Reentrant, transient
- Control Sections: IEE4703D

IEE4803D: SVC 34 -- MCS CONSOLE Information Routine

This routine constructs a message which shows the current status of the varied console.

- Entry: IEE4803D, from IEE4903D
- Exit: IEE7303D, return to caller
- Tables/Work Areas: XSA, message area, UCB, CVT, and UCM
- Attributes: Reentrant, transient
- Control Sections: IEE4803D

IEE4903D: SVC 34 -- VARY CONSOLE Processor Routine

This module processes the VARY CONSOLE command.

- Entry: IEE4903D, from IEE4203D
- Exit: To IEE4803D to construct console message
- Tables/Work Areas: XSA, CVT, UCB, and UCM
- Attributes: Reentrant, self-relocating, read-only, transient
- Control Sections: IEE4903D

IEE5403D: SVC 34 -- Command Translator Routine

This routine translates lower case letters (except those within apostrophes) into upper case letters.

- Entry: IEE5403D
- Exit: IEE0403D
- Tables/Work Areas: CVT, Internal Translation Table, UCM, UCME, XSA
- Control Section: IEE5403D

IEE5503D: SVC 34 -- STOPMN Command Handler Routine (Load 2)

This routine terminates action initiated by the MONITOR command.

- Entry: IEE5503D from IEE4503D
- Exits: IEE6703D, or return to caller
- Tables/Work Areas: XSA, M/S resident data area, CVT, UCM
- Attributes: Reentrant, self-relocating, read-only, transient
- Control Section: IEE5503D

IEE5603D: SVC 34 -- MSGRT and CONTROL Message Module (Load 1)

This routine issues error messages resulting from routing location (MSGRT or L = caa) errors and errors involving CONTROL command.

- Entry: IEE5603D
- Exit: IEE5903D, return to caller
- Tables/Work Areas: DCM, UCM, XSA
- Attributes: Reentrant, transient, type 4 SVC

IEE5703D: SVC 34 -- VARY Hardcopy OFF and Message Routing Routine

This routine removes the hardcopy log and routes any error messages to the appropriate error message routine.

- Entry: IEE5703D, from IEE4703D
- Exit: IEE0503D or IEE2103D
- Tables/Work Areas: XSA, UCM, UCB, CVT, MRC
- Attributes: Reenterable, transient ;
- Control Section: IEE5703D

IEE5903D: SVC 34 -- MSGRT and CONTROL Message Module (Load 2)

This routine issues error messages resulting from routing location (L = cca), message routing defaults (MSGRT command), or CONTROL command errors. This module completes processing that was begun by IEE5603D.

- Entry: IEE5903D from IEE5603D
- Exit: Return to caller
- Attributes: Reentrant, transient, type 4 SVC
- Tables/Work Areas: DCM, UCM, XSA

IEE60110: Master Scheduler -- DUMP Command Processor Routine

This routine handles DUMP commands entered from the operator's console. The DUMP command provides a core image dump to the SYS1.DUMP data set.

- Entry: IEE60110
- Exit: IEE7903D
- Attributes: Reentrant
- Control Section: IEE60110

IEE6303D: SVC 34 -- MSGRT Command Routine (Load 1)

This routine processes default routing specifications entered by means of the MSGRT command (with the exception of MSGRT REF, which is handled by IEE6403D).

- Entry: IEE6303D
- Exit: XCTL to IEE6403D or return to caller
- Tables/Work Areas: UCM, XSA, MRCT
- Attributes: Reentrant, transient, type 4 SVC
- CSECT: IEE6303D

IEE6403D: SVC 34 -- MSGRT Command Routine (Load 2)

This routine processes the MSGRT REF command and constructs the required display.

- Entry: IEE6403D from IEE6303D
- Exits: IEE5603D or return to caller
- Attributes: Reentrant, transient, type 4 SVC
- Tables/Work Areas: UCM, XSA, DCM, RCT Message Routing Routine

IEE6503D: SVC 34 -- SET Time-Of-Day (TOD) Clock Routine, Part I

This routine performs processing for the TOD clock for System/370 models. It calculates the discrepancy between the currently indicated time and the new time being set; it also calculates the time at midnight to maintain the correct date and then passes these values to IEE6603D.

- Entry: IEE6503D
- Exit: IEE6603D
- Tables/Work Areas: CVT, M/S Resident Data Area, XSA
- Attributes: Reentreable, transient, disabled for system interrupts
- Control Section: IEE6503D

IEE6603D: SVC 34 -- SET Time-of Day (TOD) Clock Routine, Part II

This routine sets the TOD clock, updates the CVT, and updates TSO and system TQE's, using values passed from IEE6503D. It then issues a message indicating that the SET command has been processed.

- Entry: IEE6603D
- Exit: Return to issuer of SVC 34
- Tables/Work Areas: CVT, XSA, M/S resident data area
- Attributes: Reenterable, transient, disabled for system interrupts
- Control Section: IEE6603D

IEE6703D: SVC 34 -- CONTROL Command Handler (Load 1)

This routine performs syntax check and processing for STOPMN; CONTROL E; CONTROL D; CONTROL S; CONTROL V; and CONTROL N. It passes control to other modules for CONTROL A; CONTROL M; CONTROL C. It also passes control to an error message module (IEE5603D) if an error is encountered.

- Entry: IEE6703D from IEE7503D
- Exits: IEE7703D, IEE7803D, IEE5603D or return to caller.
- Attributes: Refreshable, privileged, type 4 SVC
- Tables/Work Areas: XSA, DCM

IEE6803D: SVC 34 -- CONTROL Command Handler (Load 2)

This routine processes CONTROL A command (except CONTROL A, REF, which is processed by IEE6903D).

- Entry: IEE6803D from IEE7803D
- Exit: IEE5603D or return to caller
- Attributes: Refreshable, privileged, type 4 SVC
- Tables/Work Areas: XSA, CVT, UCM, UCME, DCM SACB ID list (this table is constructed by this module)

IEE6903D: SVC 34 -- CONTROL Command Handler (Load 5)

This routine handles CONTROL A, REF commands.

- Entry: IEE6903D from IEE7803D
- Exits: IEE5603D or return to caller
- Attributes: Refreshable, privileged, type 4 SVC
- Tables/Work Areas: DCM, UCM, XSA

IEE7103D: SVC 34 -- MONITOR Command Router

This routine identifies the MONITOR command and routes control to the appropriate routine for further processing.

- Entry: IEE7103D from IEE3503D
- Exit: IEE0503D, IEE2103D, IEE7503D, or return to caller
- Attributes: Reentrant, transient
- Tables/Work Areas: XSA, CVT, UCM, M/S Resident Data Area

IEE7203D: SVC 34 -- VARY HARDCOPY UNIT Processor

This routine process HARDCOPY or UNIT operands of the VARY command.

- Entry: IEE7203D from IEE4703D
- Exit: IEE4103D, IEE5703D
- Attributes: Reentrant, transient, privileged
- Tables/Work Areas: CVT, UCM, XSA, UCB, M/S Resident Data Area

IEE7303D: SVC 34 -- MCS Console Information (Load 2)

This routine constructs messages displaying console configuration resulting from VARY commands.

- Entry: IEE7303D from IEE4803D
- Exits: IEE4803D or return to caller
- Attributes: Reentrant, transient
- Tables/Work Areas: XSA, UCB, UCM, UCME, Buffer Core, DCM (including the SACBs)

IEE7503D: SVC 34 -- Routing Location Routine (Load 1)

This routine processes location operand (L =cca) and message routing defaults for CONTROL, STOPMN, DISPLAY and MONITOR commands:

- Entry: IEE7503D from IEE3503D, IEE7103D, IEE4503D, IEE0403D
- Exits: IEE6703D, IEE7603D, IEE5603D
- Tables/Work Areas: XSA, RCT, UCM, Resident DCM (including SACBs), M/S Resident Data Area
- Attributes: Reentrant, refreshable, privileged, type 4 SVC
- CSECT: IEE7503D

IEE7603D: SVC 34 -- Routing Location Module (Load 2)

This routine continues processing of the routing location operands (L = cca) of the DISPLAY, MONITOR or STOPMN commands. This processing starts with module IEE7503D.

- Entry: IEE7603D from IEE7503D
- Exits: IEE0803D, IEE5503D, or return to caller
- Attributes: Reentrant, refreshable, privileged, type 4 SVC
- Tables/Work Areas: XSA, UCM, Resident DCM (including the SACBs), M/S Resident Data Area

IEE7703D: SVC 34 -- CONTROL Command Handler (Load 3)

This routine processes CONTROL M commands.

- Entry: IEE7703D from IEE6703D
- Exit: IEE5603D or return to caller

- Attributes: Refreshable, privileged, type 4 SVC
- Tables/Work Areas: UCM, WQE, XSA

IEE7803D: SVC 34 -- CONTROL Command Handler (Load 4)

This routine processes CONTROL C commands.

- Entry: IEE7803D from IEE6703D
- Exit: IEE5603D, IEE6803D, IEE6903D or return to caller
- Attributes: Reentrant, transient, type 4 SVC
- Tables/Work Areas: XSA, UCM, WQE

IEE7903D: SVC 34 -- Message Module

This routine assembles and edits messages for the DUMP Command Processor, and writes the message to the operator.

- Entry: IEE7903D
- Exit: Return to caller
- Attributes: Reentrant
- Control Section: IEE7903D

IEE8503D: SVC -- DISPLAY SQA Routine

This routine displays the low and high boundaries of the SQA and the amount of free storage within the SQA.

- Entry: IEE8503D from IEE3503D
- Exit: Return to caller
- Tables/Work Areas: M/S Resident Data Area, XSA, CVT
- Attributes: Reentrant
- Control Section: IEE8503D

IEE8603D: SVC 34 -- SET Commcnd Routine, Part II

This module processes the SET command parameters: Q, PROC, and AUTO.

- Entry: IEE8603D
- Exit: To IEE0903D, IEE6503D, IEE0503D, or return to caller
- Tables/Work Areas: CVT, Master Scheduler Resident Data Area, XSA
- Attributes: Reentrant, self-relocating, read-only, transient
- Control Section: IEE8603D

IEFACTFK: Termination -- User Dummy Accounting Routine

This routine takes the place of the user's accounting routine when a user accounting routine was specified at system generation, but none was supplied.

- Entry: IEFACTLK
- Exit: Return to caller
- Control Section: IEFACTLK

IEFACTLK: Termination -- User Accounting Routine Linkage Routine

This routine provides linkage between the termination routine and the user's accounting routine. It also sets up the required parameter list -- including the execution time of the job step -- and reads the first record of the account control table.

- Entry: IEFACTLK
- Exits: To user's accounting routine, return to caller.
- Tables/Work Areas: LCT, JCT, SCT, JACT, SACT, QMPA
- Control Section: IEFACTLK

IEFACTRT: Termination -- Dummy Accounting Routine

This routine takes the place of the user-supplied accounting routine.

- Entry: IEFACTRT
- Exit: Return to caller
- Control Section: IEFACTRT

IEFAVFAK: I/O Device Allocation -- Linkage to IEFXV001

This routine passes control to the AVR routine (IEFXV001) via and XCTL macro instruction.

- Entry: IEFXV001
- Exit: XCTL to IEFXV001
- Control Section: IEFXV001

IEFCVFAK: I/O Device Allocation -- Linkage to IEFMCVOL

This routine passes control to Mount Control-Volume Routine IEFMCVOL via an XCTL

macro instruction to one of three entry points, IEFCVOL1, IEFCVOL2, or IEFCVOL3.

- Entries: IEFCVOL1, IEFCVOL2, IEFCVOL3
- Exits: XCTL to IEFCVOL1, IEFCVOL2, IEFCVOL3
- Control Section: IEFCVOL1

IEFDSDRP: Data Set Descriptor Record Processing Routine

This routine processes the job queue information in the DSDR record to make a restarting job's queue entry reflect the environment when the checkpoint was taken.

- Entry Point: IEFDSDRP
- Exit: Return to caller
- Tables/Work Areas: JCT, SCT, SIOT, JFCB, TIOT, UCB, CVT, VOLT, TCB, QMPA, CSCB, DCBD, DCB, JFCBX, SCTX, LCT
- Attributes: Reenterable
- Control Section: IEFDSDRP

IEFDSOAL: Direct System Output -- SIOT and JFCB Modification

This routine modifies the SYSOUT SIOTs and JFCBs of steps that will use DSO.

- Entry: IEFDSOAL from IEFVMLS1
- Exit: IEFVMLS1 or IEFQBVMS
- Tables/Work Areas: CVT, LCT, QMPA, DSOCB, JCT, SCT, SIOT, JFCB, PIB, UCB
- Attributes: Reenterable
- Control Sections: IEFDSOAL

IEFDSOCP: Direct System Output -- Initialization Routine

This routine initializes DSO by constructing the DSOCB and performing DSO housekeeping.

- Entry: IEFDSOCP
- Exit: Return to caller
- Tables/Work Areas: TCB, DSOCB, TIOT, JFCB, UCB, CVT, ECB/IOB, QMPA, PIB
- Attributes: Reenterable
- Control Sections: IEFDSOCP

IEFDSOFB: Direct System Output -- Release DSOCB Routine

This routine frees DSOCBs allocated to a job.

- Entry: IEFDSOFB from IEFSD168, or IEFSD518
- Exit: Return to caller
- Tables/Work Areas: LCT, DSOCB, CVT, PIB, M/S resident data area, JCT
- Attributes: Reenterable
- Control Sections: IEFDSOFB

IEFDSOSM: Direct System Output -- Stop and Modify Command Processing Routine

This routine processes Stop and Modify commands for DSO.

- Entry: IEFDSOSM
- Exit: SVC 3
- Tables/Work Areas: CVT, JCB, DSOCB, UCB, PIB, CSCB, JCT, ECB/IOB, QMPA
- Attributes: Reenterable
- Control Sections: IEFDSOSM

IEFDSOWR: Direct System Output -- System Messages and Job Separator Writer

This routine writes job separators and system messages to the assigned DSO device.

- Entry: IEFDSOWR from IEFSD512 or IEFSD31Q
- Exit: IEFSD512, IEFSD31Q, IEFQMRAW, IEFSD094 or the user's separator program
- Tables/Work Areas: TCB, LCT, QMPA, DSOCB, JCT, SCT, PIB, UCB, CVT, TIOT, DCB, SMB
- Attributes: Reenterable
- Control Sections: IEFDSOWR

IEFIDMPM: Termination -- Message Module

This routine contains the messages used by the Indicative Dump routine.

- Entry: IEFIDMPM
- Attributes: Non-executable
- Control Section: IEFIDMPM

IEFK1MSG: I/O Device Allocation -- Message Module

This routine contains the messages issued by volume attribute setting routine IEFPRES.

- Entry: IEFK1MSG
- Attributes: Non-executable
- Control Section: IEFK1MSG

IEFLOCDQ: Queue Management -- Dequeue by Jobname Routine

This routine searches a queue for a named job or list of named jobs, and can return information, or dequeue or cancel the job.

- Entry: LOCDQ, LOCCAN, LOC
- Exit: Return to caller
- Tables/Work Areas: QCR, LTH
- Attributes: Reenterable
- External References: IEFCNVRT, IEFRDWRT

IEFMCVOL: I/O Device Allocation -- Mount Control-Volume Routine

This routine will have a control volume mounted when a data set called for in a job step cannot be located on any currently mounted control volume.

- Entries: IEFCVOL1, IEFCVOL2, IEFCVOL3
- Exits: IEFVM1, IEFVMCVL, IEFVM6, IEFYN (IEFW41SD)
- Tables/Work Areas: LCT, JCT, SCT, SIOT, JFCB, VOLT, QMPA, UCB
- Attributes: Reusable
- Control Sections: IEFCVOL1, IEFCVOL2, IEFCVOL3

IEFORMAT: Queue Management -- Queue Formatting Routine

This routine places the work queue data set in the format required by the MFT queue management routines.

- Entry: IEFORMAT, from IEFSD055
- Exit: Return to IEFSD055
- Tables/Work Areas: DCB, DEB
- Attributes: Reusable
- Control Section: IEFORMAT

IEFPPGM: Initiator -- Problem Program Initialization Routine

This routine prepares for the initiation of a problem program specified in a START command.

- Entry: IEFPPGM
- Exit: XCTL to IEFSD511
- Tables/Work Areas: CSCB, JCT, LOT, PIB, SPIL, TCB
- Attributes: Read-only, reenterable
- Control Section: IEFPPGM

IEFPRES: I/O Device Allocation -- Volume Attribute Setting Routine

This routine sets the volume attributes of all volumes listed in the PRESRES data set and of all unlisted, but permanently resident, direct access volumes.

- Entry: IEFPRES
- Exit: To IEEVPRES, IEFSD569, or IEFK1MSG
- Tables/Work Areas: UCB
- Control Section: IEFPRES

IEFPRINT: Direct System Output -- Tape to Printer or Card Punch Routine

This routine writes a DSO - written tape to a printer or card punch.

- Entry: SPRINTER
- Exit: Return to caller
- Control Sections: SPRINTER

IEFQAGST: Queue Management -- Assign/Start Routine

This routine sets up an ECB/IOB and prepares the queue manager parameter area for the assign routine.

- Entry: IEFQAGST
- Exit: Return to caller
- Tables/Work Areas: Q/M resident data area, QMPA, CVT
- Attributes: Reenterable
- Control Section: IEFQAGST

IEFQASGQ: Queue Management -- Assign Routine

This routine assigns records to a queue entry and assigns logical tracks as required.

- Entry: IEFQASGN
- Exit: Return to caller
- Tables/Work Areas: Q/M resident data area, QMPA, CVT
- Attributes: Reenterable
- Control Sections: IEFQASGN, IEFQASNM

IEFQBVMS: Queue Management -- Control Routine

This routine inspects the function code in the queue manager parameter area and, on the basis of this code, branches to the appropriate queue management routines.

- Entry: IEFQMSSS
- Exits: To IEFQAGST, IEFQMRAW, IEFQMNQQ, or IEFQASGQ, return to caller
- Tables/Work Areas: QMPA
- Attributes: Reenterable
- Control Section: IEFQMSSS

IEFQDELQ: Queue Management -- Delete Routine

This routine makes those logical tracks assigned to a queue entry available for assignment to other queue entries.

- Entry: IEFQDELE
- Exit: Return to caller resident data area, CVT
- Attributes: Reenterable
- Control Section: IEFQDELE

IEFQMDQQ: Queue Management -- Dequeue Routine

This routine removes the highest priority entry from an input queue or a system output queue.

- Entry: IEFQMDQ2
- Exit: Return to caller
- Tables/Work Areas: CVT, Q/M resident data area, QCR, LTH
- Attributes: Reenterable
- Control Section: IEFQMDQ2

IEFQMDUM: Queue Management -- Dummy Module

This routine prevents the occurrence of an unresolved external reference to module IEFQMSSS during system generation.

- Entry: IEFQMDUM
- Attributes: Non-Executable
- Control Section: IEFQMSSS

IEFQMLK1: Queue Management -- Branch Routine

This routine branches to the appropriate queue management routine on the basis of an assign or read/write function code issued by an initiator.

- Entry: IEFQMSSS
- Exits: To IEFQASGQ or IEFQMRAW
- Tables/Work Areas: QMPA
- Attributes: Reenterable
- Control Section: IEFQMSSS

IEFQMNQQ: Queue Management -- Enqueue Routine

This routine places an entry in an input queue or an output queue at the requested priority.

- Entry: IEFQMNQ2
- Exit: Return to caller
- Tables/Work Areas: CVT, Q/M resident data area, QMPA, QCR, LTH
- Attributes: Reenterable
- Control Section: IEFQMNQ2

IEFQMRAW: Queue Management -- Read/Write Routine

This routine performs the conversion of a TTR into a MBBCCHHR and reads or writes up to 15 records of the work queue data set.

- Entry: IEFQMRAW
- Exit: Return to caller
- Tables/Work Areas: Q/M resident data area, QMPA, CVT, IOB/ECB
- Attributes: Reenterable

- Control Section: IEFQMRAW

IEFQMUNQ: Queue Management -- Unchain Routine

This routine removes a task from the queue management no-work chain.

- Entry: IEFQMUNC
- Exit: Return to caller
- Tables/Work Areas: CVT, Q/M resident data area, QCR
- Attributes: Reenterable
- Control Section: IEFQMUNC

IEFQRESD: Queue Management -- Resident Main Storage Reservation Module

This routine reserves 140 bytes of resident main storage for the queue-management-opened DCB/DEB and the master queue control record at nucleus initialization time.

- Attributes: Non-executable
- Control Section: IEFJOB

IEFRCLN1: Restart Reader Linkage

This routine receives control from IEFVRRC and LINKS to interpreter initialization routine IEFVH1.

- Entry: IEFRCLN1
- Exit: XCTL to IEFVRRC at entry IEFVRRCA
- Attributes: Reenterable
- Control Section: IEFRCLN1

IEFRCLN2: Restart Reader Linkage

This routine receives control from IEFVRRC and LINKS to interpreter initialization routine IEFVH1.

- Entry: IEFRCLN2
- Exit: XCTL to IEFVRRC at entry IEFVRRCB
- Attributes: Reenterable
- Control Section: IEFRCLN2

IEFRPREP: Termination -- Restart Preparation Routing

This routine determines whether a job step that has been abnormally terminated can be restarted.

- Entry: IEFRPREP from IEFYNIMP
- Exit: Return to caller
- Attributes: Reenterable
- Tables/Work Areas: LCT, JCT, SCT, PDQ, QMPA
- Control Section: IEFRPREP

IEFRSTRT: Restart SVC Issuing Routine

This routine issues the Restart SVC. When called by its alias, IEFSMR, it issues the Restart SVC and then returns to the caller.

- Entry: IEFRSTRT, IEFSMR
- Exit: SVC 52 (RESTART), return to caller
- Attributes: Reenterable
- Control Sections: IEFRSTRT

IEFSDXXX: System Output Writer -- Spanned Data Sets Handler

This routine handles variable-length data sets by dynamically moving the input segments into the output segments.

- Entry: IEFSDXXX
- Exit: Return to caller
- Tables/Work Areas: Control area for spanning, DCB
- Attributes: Reentrant
- Control Section: IEFSDXXX

IEFSDXYZ: System Output Writer -- Command Chaining Access Method

This routine simulates the processing performed by the QSAM Put Locate and Truncate routines by using command chaining to write one string of up to nine buffers with a single IOB and a single EXCP macro instruction.

- Entry: IEFSDXYZ
- Exit: Return to caller
- Tables/Work Areas: ECB, IOB, I/O table
- Attributes: Reentrant
- Control Section: IEFSDXYZ

IEFSD017: Termination -- System Output Interface Routine

This routine provides an interface between the termination entry routine and system output processing.

- Entry: IEFSD017
- Exit: To IEFSD42Q
- Control Section: IEFSD017

IEFSD055: Queue Management -- Queue Initialization Routine

This routine constructs a resident DEB/DCB, passes control to the queue formatting routine or the first phase of system restart, initializes the queue manager resident data area, and (if required) passes control to the second phase of the system restart routine.

- Entry: IEFSD055, from IEFQINTZ
- Exits: To IEFORMAT, IEF300SD, or IEF304SD
- Attributes: Reusable
- Control Section: IEFSD055

IEFSD070: System Output Writer -- Data Set Writer Interface Routine

This routine passes control to the standard data set writer or to the user-supplied data set writer routine.

- Entry: IEFSD070
- Exits: To IEFSD087 or user-supplied routine via LINK, or to IEFSD171 via XCTL
- Attributes: Reenterable
- Control Section: IEFSD070

IEFSD078: System Output Writer -- Linker Routine

This routine determines whether the record obtained from the output queue entry is a DSB or SMB, and passes control, accordingly, to the DSB or SMB processor.

- Entry: IEFSD078
- Exits: To IEFSD085, IEFSD086, or IEFSD079
- Attributes: Reenterable
- Control Section: IEFSD078

IEFSD079: System Output Writer -- Link to Queue Manager Delete Routine

This routine passes control to the delete routine to delete the current output queue entry.

- Entry: IEFSD079
- Exits: To IEFQDELQ and IEFSD082
- Tables/Work Areas: QMPA
- Attributes: Reenterable
- Control Section: IEFSD079

IEFSD080: System Output Writer -- Initialization Routine

This routine initializes the system output writer by obtaining main storage for a parameter list and the output DCB, and opening the output DCB.

- Entry: IEFSD080
- Exit: To IEFSD081
- Tables/Work Areas: DCB, CSCB, TIOT, JFCB
- Attributes: Reenterable
- Control Section: IEFSD080

IEFSD081: System Output Writer -- Class Name Setup Routine

This routine obtains main storage for, and initializes, a list of ECB pointers, ECBs, and queue management communication elements, depending on the system output classes specified for the writer.

- Entry: IEFSD081
- Exit: To IEFSD082
- Tables/Work Areas: CSCB, ECB
- Attributes: Reenterable
- Control Section: IEFSD081

IEFSD082: System Output Writer -- Main Logic Routine

This routine obtains main storage for QMPAs and internal work areas, dequeues output queue entries, checks for operator commands, and passes control to the appropriate routine.

- Entry: IEFSD082
- Exits: IEFSD083, IEFSD084, IEFSD078

- Tables/Work Areas: CSCB, ECB
- Attributes: Reenterable
- Control Section: IEFSD082

IEFSD083: System Output Writer -- Command Processing Routine

This routine processes MODIFY and STOP commands that apply to the writer.

- Entry: IEFSD083
- Exits: To IEFSD081 or IEEVTCTL
- Tables/Work Areas: CSCB, DCB, QMPA, ECB
- Attributes: Reenterable
- Control Sections: IEFSD083, IEFSD83M

IEFSD084: System Output Writer -- Wait Routine

This routine waits for an entry to be enqueued in an output queue corresponding to a class available to the writer.

- Entry: IEFSD084
- Exit: To IEFSD082
- Attributes: Reenterable
- Control Section: IEFSD084

IEFSD085: System Output Writer -- DSB Handler Routine

This routine initializes for data set processing, and informs the operator of the pause option in effect.

- Entry: IEFSD085, IEF085SD, or IEF850SD
- Exit: To IEFSD171
- Attributes: Reenterable
- Control Sections: IEFSD085, IEFSD85M

IEFSD086: System Output Writer -- SMB Handler

This routine initializes for message processing, and extracts each message from the current SMB.

- Entry: IEFSD086, IEF086SD
- Exits: To IEFSD088, IEFSD089, IEFQMNQQ, IEFQMRAW, IEFSD085, IEFSD078
- Tables/Work Areas: SMB, UCB, QMPA, TIOT, CSCB, TCB
- Attributes: Reenterable
- Control Sections: IEFSD086, IEFSD86M

IEFSD087: System Output Writer -- Standard Writer Routine

This routine gets records from a data set.

- Entry: IEFSD087
- Exits: To IEFSD088, IEFSD089, IEFSD078
- Tables/Work Areas: DCB
- Attributes: Reenterable
- Control Sections: IEFSD087, IEFSD87M

IEFSD088: System Output Writer -- Transition Routine

This routine handles the transition between messages and data sets, and between data sets.

- Entry: IEFSD088
- Exit: To IEFSD089
- Tables/Work Areas: DCB
- Attributes: Reenterable
- Control Section: IEFSD088

IEFSD089: System Output Writer -- Put Routine

This routine formats records as required and issues PUT macro instructions to write them on the output unit.

- Entry: IEFSD089
- Exit: To IEFSD088
- Tables/Work Areas: DCB
- Attributes: Reenterable
- Control Sections: IEFSD089, IEFSD89M

IEFSD094: System Output Writer -- Job Separator Routine

This routine prints or punches a job name and system output class designation on the writer's output device.

- Entry: IEFSD094
- Exits: To IEFSD088, IEFSD089, IEFSD095, IEFSD078
- Attributes: Reenterable
- Control Section: IEFSD094

IEFSD095: System Output Writer -- Print Line Routine

This routine constructs the block letters used to separate jobs processed by a system output writer when the output data set is to be printed.

- Entry: IEFSD095
- Exit: Return to caller
- Attributes: Reenterable
- Control Section: IEFSD095

IEFSD096: System Output Writer -- Message Module

This routine contains message headers and texts for messages to the operator.

- Entry: IEFSD096
- Attributes: Non-executable
- Control Section: IEFSD096

IEFSD097: I/O Device Allocation -- Wait for Space Decision Routine

This routine makes the decision whether to wait for direct access space, and provides an interface with the I/O device allocation space request routine so that retry and additional recovery passes may be made.

- Entry: IEFSD097
- Exit: Branch on register 14
- Tables/Work Areas: LCT, TIOT, UCB
- Attributes: Read-only, reenterable
- Control Section: IEFSD097

IEFSD167: Initiator -- Linkage to IEFSD168

This routine passes control via an XCTL macro instruction to IEFSD168 in the 30K scheduler.

- Entry: IEFSD068
- Exit: IEFSD168
- Tables/Work Areas: None
- Attributes: Reenterable
- Control Section: IEFSD068

IEFSD168: Initiator -- Job Suspension

This routine causes a terminated job to be reenqueued so that the job can be reactivated.

- Entry: IEFSD068
- Exit: Branch to IEFSD598 to purge resources, branch to IEFSD510 to reinitiate job, link to IEFDSOFB
- Tables/Work Areas: QMPA, LCT, JCT, SCD, SCT
- Attributes: Reenterable
- Control Section: IEFSD068
- External Reference: IEFQMRAW, IEFQMNQ2, IEFVSDRA

IEFSD171: System Output Writer -- Data Set Delete Routine

This routine obtains records from an output queue entry, and deletes system output data sets.

- Entry: IEFSD071
- Exits: To IEEVLOUT, IEFQMNQ2, IEF850SD, IEF086SD, IEFSD078, or IEFQMRAW
- Tables/Work Areas: DCB, SMB, UCB, CVT, QMPA, TIOT, CSCB, TCB
- Attributes: Reenterable
- Control Sections: IEFSD071, IEFSD71M

IEFSD195: I/O Device Allocation -- Wait for Deallocation Routine

This routine provides the I/O device allocation routine with the ability to wait for deallocation to occur during the execution of another task, when allocation cannot be completed because of current allocations.

- Entry: IEFVAWAT
- Exit: Return to caller
- Tables/Work Areas: JCT, SCT, SIOT, LCT, ECG, CSCB
- Attributes: Read-only, reenterable
- Control Section: IEFSD095

IEFSD21Q: I/O Device Allocation -- Allocation Entry Routine

This routine provides an interface for entry to the I/O device allocation routine operating in a multiprogramming environment.

- Entry: IEFW21SD
- Exits: To IEFVK, IEFVM or IEFWD000
- Tables/Work Areas: JCT, LCT, SCT, SMB, QMPA, CVT
- Attributes: Read-only, reenterable
- Control Section: IEFWLISD

IEFSD22Q: Termination Routine -- Step Terminate Exit Routine

This routine provides an interface between the termination routine and the step delete or alternate step delete routine when a step has been terminated.

- Entry: IEFW22SD
- Exit: Return to caller of termination routine
- Tables/Work Areas: JCT, SCT, SMB, LCT, QMPA, ECB
- Attributes: Read-only, reenterable
- Control Section: IEFW22SD

IEFSD300: System Restart -- Initialization Routine

This routine reads all QCRs and logical track header records into main storage, builds tables A, B, and C, and removes from Table A all the LTH entries corresponding to logical tracks in the free-track queue or in one of the other queues.

- Entry: IEFSD300
- Exit: To IEFSD301
- Tables/Work Areas: System restart work area, Table A, Table B, Table C
- Attributes: Reenterable
- Control Section: IEFSD300

IEFSD301: System Restart -- Purge Queue Construction Routine

This routine searches Table A for the last LTH corresponding to each queue entry, determines the type of entry, and constructs the purge queue.

- Entry: IEFSD301
- Exit: To IEFSD302
- Tables/Work Areas: System restart work area, Table A, Table C purge queue, interpreter jobnames table
- Attributes: Reenterable
- Control Section: IEFSD301

IEFSD302: System Restart -- Jobnames Tables Routine

This routine removes from Table A all logical tracks assigned to dequeued input or RJE queue entries, and builds a table of jobnames for incomplete input and RJE queue entries and dequeued input queue entries.

- Entry: IEFSD302
- Exit: To IEFSD303
- Tables/Work Areas: System restart work area, Table A, Table C, and the interpreter/initiator jobnames table
- Attributes: Reenterable
- Control Section: IEFSD302

IEFSD303: System Restart -- Delete Routine

This routine creates a queue entry of the remaining logical tracks and deletes that entry, thus assigning those tracks to the free-track queue.

- Entry: IEFSD303
- Exit: Return to caller
- Tables/Work Areas: System restart work area, QMPA, Table A
- Attributes: Reenterable
- Control Section: IEFSD303

IEFSD304: System Restart -- Scratch Data Sets Routine

This routine informs the operator of the names of jobs being processed by an interpreter, and scratches temporary data sets generated for incomplete input queue entries.

- Entry: IEFSD304
- Exits: To IEFSD055, IEFSD308
- Tables/Work Areas: CVT, UCB address look-up table
- Attributes: Reenterable
- Control Section: IEFSD304

IEFSD305: System Restart -- Reenqueue Routine

This routine dequeues the entries in the purge queue and reenqueues them in the appropriate input or output queue and informs the operator of the names of jobs in the process of initiation.

- Entry: IEFSD305
- Exit: IEFSD304
- Tables/Work Areas: System restart work area, purge queue, JCT, SCT, JFCB, DSB, SCD, SIOT.
- Attributes: Reenterable
- Control Section: IEFSD305

IEFSD308: System Restart -- Scratch Data Sets Routine

This routine scratches the temporary data sets generated for incomplete input queue entries.

- Entry: IEFSD308
- Exit: Return to caller
- Tables/Work Areas: DSCB, DCB, UCB, CVT, VTOC, DEB
- Attributes: Reenterable
- Control Section: IEFSD308

IEFSD31Q: Termination Routine -- Job Termination Exit Routine

This routine provides an interface between the termination routine and the step delete or alternate step delete routine when the last step of a job has been terminated. If DSO was used, the DSOCBs are released; if the message class is assigned to DSO, the routine links to IEFDSOWR.

- Entry: IEFW31SD
- Exit: IEFSD32Q (44K Scheduler), IEFSD33Q (30K Scheduler), or IEFDSOWR
- Tables/Work Areas: JCT, SCT, SMB, QMPA, ECB, CVT, M/S resident data area, DSOCB, PIB
- Attributes: Read-only, reenterable
- Control Section: IEFW31SD

IEFSD310: System Restart -- TTR and NN to MBBCCHHR Conversion Routine

This routine converts a relative record address (NN) or a relative track and record address (TTR) to an actual disk address (MBBCCHHR).

- Entry: IEFSD310
- Exit: Return to caller
- Tables/Work Areas: CVT
- Attributes: Reenterable
- Control Section: IEFSD310

IEFSD311: Queue Management -- Message Module

This routine contains the messages required by the queue initialization routine (IEFSD055).

- Entry: IEFSD311, SD55MSG1, SD55MSG2, SD55MSG3
- Attributes: Non-executable
- Control Section: IEFSD311

IEFSD312: System Restart -- Message Module

This routine contains the messages required by the system restart routines.

- Entry: IEFSD312, SD304MG1, SD304MG2, SD305MG1
- Attributes: Non-executable
- Control Section: IEFSD312

IEFSD32Q: Initiator -- Linkage From Job Termination to the Initiator for the 44K Scheduler

This routine receives control from job termination exit routine IEFSD31Q in MFT systems with the 44K scheduler. It returns control to step deletion routine IEFSD515 via the RETURN macro instruction.

- Entry: IEFW32SD
- Exit: Return to IEFSD515
- Tables/Work Areas: None
- Attributes: Reenterable
- Control Section: IEFW32SD

IEFSD33Q: Initiator -- Linkage From Job Termination to the Initiator for the 30K Scheduler

This routine receives control from job termination routine IEFSD31Q in MFT systems with the 30K scheduler. It passes control to job deletion routine IEFSD517 with the

address of the Life-of-Task block in register 1.

- Entry: IEFW32SD
- Exit: Branch to IEFSD517
- Tables/Work Areas: None
- Attributes: Reenterable
- Control Section: IEFW32SD

IEFSD41Q: I/O Device Allocation -- Allocation Exit Routine

This routine provides an interface for exit from the I/O device allocation routine operating in a multiprogramming environment.

- Entry: IEFW41SD, IEFW1FAK, IEFW2FAK
- Exits: To IEFVM, or return to caller
- Tables/Work Areas: ATCA, JCT, LCT, MVA, MVAX, SCT, SMB, QMPA
- Attributes: Read-only, reenterable
- Control Section: IEFW41SD

IEFSD42Q: Termination Routine -- Termination Entry Routine

This routine provides an interface for entry to the termination routine operating in a multiprogramming environment. it also provides an interface for entry to the LOG function if a LOG data set is scheduled to be added to the SYSOUT queue.

- Entry: IEFW42SD
- Exit: To IEFYN
- Tables/Work Areas: JCT, SCT, SMB, LCT, TIOT
- Attributes: Read-only, reenterable
- Control Section: IEFW42SD

IEFSD510: Initiator -- Job Selection Routine

This routine selects a system or problem program job. This module executes only in a large (scheduler-size) partition.

- Entry: IEFSD510
- Exits: Branch to IEFSD511 or IEFSD515, LINK to IEFSD519, XCTL to IEFSD586, IEFSD589, SMALTERM
- Tables/Work Areas: LOT block, CSCB, SPIL, CVT, TCB, PIB, DSOCB
- Attributes: Read-only, reenterable
- Control Section: IEFSD510
- External References: IEFQMDQQ, IEFQMUNC

IEFSD511: Initiator -- Job Initiation Routine

This routine initializes information pertaining to a job. If DSO is available for the job's system output classes, the routine selects the DSOCBs to be used by the job.

- Entry: IEFSD511, IEFDSOSL
- Exit: Branch to IEFSD541
- Tables/Work Areas: Life-of-Task Block, CSCB, JCT, SCT, SCD, PIB, IOB2, DSOCB
- Attributes: Read-only, Reenterable
- Control Section: IEFSD511, IEFDSOSL
- External References: IEFQMRAW

IEFSD512: Initiator -- Step Initiation Routine

This routine passes control to allocation as a closed subroutine via an XCTL macro instruction and receives control back from Allocation at entry point IEFALRET. If an allocation error occurs, it passes control to the Alternate Step Deletion routine. Otherwise, it continues normally and schedules a job step.

- Entry: IEFSD512, IEFALRET
- Exits: Branch to IEFSD513, IEFSD516, or IEFSD518, XCTL to IEFSD510, IEFSD556
- Tables/Work Areas: LOT Block, JCT, SCT, APL, TIOT, IOB1, IOB2, QMPA, SMB, DSOCB
- Attributes: Read-only, reenterable
- Control Section: IEFSD512
- External References: IEFQMRAW, IEFSD556, IEFSD514, IEFDSOWR

IEFSD513: Initiator -- Problem Program Interface

This routine prepares the partition for problem program execution by moving the TIOT to the highest available storage area.

The routine also opens JOBLIB and FETCH DCBs, if required. A final check is made to determine if a CANCEL command has been received for the job before the problem

program is brought into the partition and given control. If scheduling was performed for a small partition, IEFSD513 communicates with the small partition.

- Entry: IEFSD513
- Exits: XCTL to problem program, ABEND, or to IEFSD510
- Tables/Work Areas: LOT Block, Transfer Parameter List, TIOT, User's Parameter List, TCB, CVT, PIB, CSCB, SPIL, APL, JCT, SCT, DCB
- Attributes: Read-only, reenterable
- Control Section: IEFSD513

IEFSD514: Queue Management -- Figure Breakup Routine

This routine reads and writes tables which may be required by the job scheduler. The routine breaks the tables into 176-byte records, writes the records on disk, and retrieves the records from disk to reconstruct the tables in main storage.

- Entry: IEFSD514
- Exit: Return to caller
- Tables/Work Areas: QMPA, TBR Parameter List
- Attributes: Read-only, reenterable
- Control Section: IEFSD514
- External References: IEFQASGN, IEFQMRAW

IEFSD515: Initiator -- Step Deletion Routine

This routine retrieves the TIOT and Life-of-Task Block from disk, reads in the JCT and SCT, and branches to termination, which is used as a closed subroutine. It also reads in the SCT for the next step to be scheduled, if required.

- Entry: IEFSD515, SMALTERM, or GO
- Exits: XCTL to IEFSD512 or Branch to IEFSD517, IEFSD558, IEFSD167 (30K Scheduler), IEFSD168 (44K Scheduler), or BALR to IEFSD42Q
- Tables/Work Areas: Life-of-Task Block, Terminate Parameter List, CVT, TCB, PIB, IOB, CSCB, DCB, JCT, SCT, SPIL, DSOCB
- Attributes: Read-only, reenterable
- Control Section: IEFSD515
- External References: IEFQMRAW, IEFSD514, IEFSD42Q, IEFSD598

IEFSD516: Initiator -- Alternate Step Deletion Routine

This routine provides an interface with termination when an allocation error occurs during step initiation. Termination is used as a closed subroutine. If required, this routine reads the SCT of the next step to support job flushing.

- Entry: IEFSD516
- Exits: Branch to IEFSD512 or IEFSD517
- Tables/Work Areas: Life-of-Task block, CSCB, Terminate Parameter List, SCT
- Attributes: Read-only, reenterable
- Control Section: IEFSD516
- External References: IEFQMRAW, IEFSD42Q

IEFSD517: Initiator -- Job Deletion Routine

This routine deletes the disk queue entry for a terminated job and unchains and deletes the CSCB for the job.

- Entry: IEFSD517
- Exit: Branch to IEFSD510
- Tables/Work Areas: CSCB, Life-of-Task block, SPIL
- Attributes: Read-only, reenterable
- Control Section: IEFSD517
- External References: IEFQDELE, IEFSD598

IEFSD518: Initiator -- Partition Recovery Routine

This routine determines the status of main storage required for a checkpoint/restart.

- Entry: IEFSD518
- Exits: Return to IEFSD512
- Tables/Work Areas: SPIL, CVT, TCB, JCT, PIB, LOT, QMPA, CSCB, DSOCB
- Attributes: Reenterable
- Control Section: IEFSD518

- External Reference: IEFQMRAW, IEFQMNQ2, IEFSD598, IEFDSOFB

IEFSD519: Queue Management -- Dequeue by Jobname Interface Routine

This routine builds a seven-word parameter list used by IEFLOCDQ to locate a job by jobname on the checkpoint/restart internal queue.

- Entry: IEFSD519
- Exit: Return to IEFSD510
- Tables/Work Areas: LOT, PIB
- Attributes: Reenterable
- Control Section: IEFSD519
- External Reference: IEFLOCDQ, IEFQMRAW

IEFSD530: Interpreter -- Transient Reader Suspend Routine

This routine closes the reader input data set and procedure library, and saves data required to restore the reader.

- Entry: IEFSD530
- Exit: Return to caller
- Tables/Work Areas: IWA, TIOT, LWA, QMPA, CVT, UCB, MSRC, PIB, CSCB
- Attributes: Read-only, reenterable
- Control Section: IEFSD530
- External References: IEFSD514, IEFQMRAW, IEFQASNM, IEFQASGN

IEFSD531: Interpreter -- Transient Reader Restore Routine

This routine restores the information required to "restart" a transient reader after it has been suspended. It reopens the reader input data set and procedure library.

- Entry: IEFSD531
- Exit: XCTL to IEFVHCB
- Tables/Work Areas: IWA, TIOT, QMPA, CVT, UCB, MSRC, PIB, CSCB
- Attributes: Read-only, reenterable
- Control Sections: IEFSD531, IEFPH2
- External References: IEFSD514, IEFQMRAW, IEFQASNM, IEFQASGN

IEFSD532: Interpreter -- Transient Reader Suspend Tests

This routine determines the status of a transient reader. IEFSD532 receives control from IEFVHH after a job has been enqueued.

- Entry: IEFKG
- Exits: XCTL to IEFVHN or IEFSD530, or branch to IEFVHHB
- Tables/Work Areas: IWA, LWA, QMPA, PIB, CVT
- Attributes: Read-only, reenterable
- Control Section: IEFKG

IEFSD533: Interpreter -- Interface Routine

This routine provides an interface between the reader/interpreter and system task control.

- Entry: IEFIRC
- Exits: XCTL to IEFSD537. RETURN to STC if error.
- Tables/Work Areas: CSCB, CVT, QMPA
- Attributes: Reenterable, read-only
- Control Section: IEFIRC

IEFSD534: System Task Control -- LPSW Routine

This routine places system task control in problem program mode by loading a PSW.

- Entry: IEFSD534
- Exit: XCTL to IEEVSTAR
- Tables/Work Areas: None
- Attributes: Reenterable
- Control Section: IEFSD534

IEFSD535: System Task Control -- LPSW Routine

This routine places system task control in the problem program mode by loading a PSW.

- Entry: IEFSD535
- Exit: XCTL to IEEVTCTL
- Tables/Work Areas: None
- Attributes: Reenterable

- Control Section: IEFSD535

IEFSD536: Interpreter -- Operator Message Routine

This routine writes a message to the operator when an I/O error or CPO full condition has occurred. The routine also sets proper indicators to cause a cleanup of the current job.

- Entry: IEFVHR
- Exits: Return to caller, XCTL to IEFVHN, or LINK to IEFSD308
- Tables/Work Areas: IWA, JCT, LWA, UCB, CVT, PIB, CSCB, Master Scheduler resident data area
- Attributes: Read-only, reenterable
- Control Section: IEFVHR

IEFSD537: Interpreter -- Linkage Module

This routine provides an interface between system task control and a reader. It also frees the interpreter entrance list (NEL) and associated areas if a reader is being terminated or suspended.

- Entry: IEFSD537
- Exits: LINK to IEFVH1, or IEFSD531, or Return to system task control
- Tables/Work Areas: *NEL
- Attributes: Read-only, reenterable
- Control Section: IEFSD537

IEFSD540: Initiator -- Linkage to IEFSD541

This routine provides an interface linkage to IEFSD541 via an XCTL macro instruction.

- Entry: IEFSD540
- Exit: XCTL to IEFSD541
- Tables/Work Areas: Same as caller
- Attributes: Read-only, reenterable
- Control Section: IEFSD540

IEFSD541: Initiator -- Data Set Integrity

This routine enqueues on explicit data sets and thus prevents concurrent, and impairing, access between tasks.

- Entry: IEFSD541
- Exit: Branch to IEFSD512
- Tables/Work Areas: LOT Block, IOB1, IOB2, JCT, SCT, CSCB, SPIL, DSENQ Table, Minor Name List, ENQ supervisor list.
- Attributes: Read-only
- Control Section: IEFSD541
- External References: IEFQMRAW

IEFSD551: I/O Device Allocation -- Linkage to IEFXJIMP

This routine provides an interface linkage to IEFXJIMP via an XCTL macro instruction in the 30K design package.

- Entry: IEFV15XL
- Exit: XCTL to IEFXJIMP
- Tables/Work Areas: Same as caller
- Attributes: Read-only, reenterable
- Control Section: IEFV15XL

IEFSD552: I/O Device Allocation -- Linkage to IEFXJIMP

This routine provides an interface linkage to IEFXJIMP via an XCTL macro instruction in the 30K design package.

- Entry: IEFXJX5A
- Exit: XCTL to IEFXJIMP
- Tables/Work Areas: Same as caller
- Attributes: Read-only, reenterable
- Control Section: IEFXJX5A

IEFSD553: Initiator -- Linkage to IEFSD512

This routine provides a linkage to IEFSD512 via an XCTL macro instruction.

- Entry: IEFSD512
- Exit: XCTL to IEFSD512
- Tables/Work Areas: Same as caller
- Attributes: Read-only, reenterable
- Control Section: IEFSD512

IEFSD554: Initiator -- Linkage to IEFSD516

This routine provides a linkage to IEFSD516 via an XCTL macro instruction.

- Entry: IEFSD554

- Exit: XCTL to IEFSD516
- Tables/Work Areas: Same as caller
- Attributes: Read-only, reenterable
- Control Section: IEFSD554

IEFSD555: Initiator -- Linkage to IEFSD510

This routine provides linkage to IEFSD510 via an XCTL macro instruction.

- Entry: IEFSD555
- Exit: XCTL to IEFSD510
- Tables/Work Areas: Same as caller.
- Attributes: Read-only, reenterable
- Control Section: IEFSD555

IEFSD556: Initiator -- Set Problem Program State Return

This routine establishes the allocation routine in a problem program state, upon entry.

- Entry: IEFSD556
- Exit: LPSW to IEFW21SD
- Tables/Work Areas: Same as caller.
- Attributes: Read-only, reenterable
- Control Section: IEFSD556

IEFSD557: I/O Device Allocation -- Interface Routine

This routine provides an interface between system task control and allocation.

- Entry: IEFW21SD
- Exit: IEFWSD21
- Tables/Work Areas: ECB, IOB
- Attributes: Reenterable
- Control Section: IEFSD557

IEFSD558: Initiator -- Linkage to IEFSD511

This routine provides a linkage to IEFSD511 via an XCTL macro instruction.

- Entry: IEFSD558
- Exit: IEFSD511
- Attributes: Read-only, reenterable
- Control Section: IEFSD558

IEFSD559: Initiator -- Linkage to IEFSD515

This routine provides a linkage to IEFSD515 via an XCTL macro instruction.

- Entry: SMALTERM
- Exit: IEFSD515
- Attributes: Read-only, reenterable
- Control Section: IEFSD559

IEFSD567: Nucleus -- I/O Device Allocation Device-End Interrupt Handler Routine

This routine handles the posting of unsolicited device interruptions for I/O device allocation operating in a multiprogramming environment.

- Entry: IEFDPOST
- Exits: IEAOPT01 or return to caller
- Tables/Work Areas: ATCA, MVA, MVAX
- Attributes: Disabled, resident
- Control Section: *IEFDPOST

IEFSD569: Master Scheduler -- Initialization Routine

This routine initializes the communications task and the system log. It issues the READY message and formats the job queue, as well as typing out the automatic commands and invoking processing of the automatic commands. This routine establishes partitioning of main storage at system initialization and readies the partitions for the START command. It is called out at system generation by the macro, SGIEE0VV.

- Entry: IEFSD569
- Exit: IEE0503D, Branch to dispatcher
- Attributes: Read-only, non-reenterable
- Control Section: IEFSD569

IEFSD572: Queue Management -- Interpreter/ Queue Manager Interlock Routine

This routine determines if a possible interlock condition exists between the queue manager and the reader. The routine issues a message requesting the operator to reply with either WAIT, to wait for space to be freed, or CANCEL, to cancel the job.

- Entry: IEFSD572

- Exits: ABEND, or return to caller
- Attributes: Read-only, reenterable
- Control Section: IEFSD572, IEFSD573
- External Reference: IEFQDELQ

IEFSD584: System Task Control -- LPSW Routine

This routine places system task control in the problem program mode by loading a PSW.

- Entry: IEFSD584
- Exit: XCTL to IEESD591
- Control Section: IEFSD584

IEFSD585: System Task Control -- LPSW Routine

This routine places the DSO processor in the problem program mode by loading a PSW.

- Entry: IEFSD585
- Exit: XCTL to IEFDSOSM
- Control Section: IEFSD585

IEFSD586: System Task Control -- Linkage to IEFSD585

This routine links to IEFSD585 so that upon return the initiator will be in supervisor state.

- Entry: IEFSD586
- Exit: Link to IEFSD585
- Control Section: IEFSD586

IEFSD587: System Task Control -- Linkage to IEFSD535

This routine provides a linkage to IEFSD535 via a LINK macro instruction, so that upon return the initiator will be in the supervisor state.

- Entry: IEFSD587
- Exit: Link to IEFSD535, XCTL to IEFSD510
- Attributes: Read-only, reenterable
- Control Section: IEFSD587

IEFSD588: System Task Control -- Linkage to IEE591SD

This routine links to IEE591SD to bring the suspended reader into the assigned partition so that upon return, the initiator will be in supervisor state.

- Entry: IEFSD588
- Exit: XCTL to IEFSD510
- Tables/Work Areas: Same as caller
- Attributes: Read-only, reenterable
- Control Section: IEFSD588

IEFSD589: System Task Control -- Linkage to IEFSD534

This routine links to IEFSD534 so that upon return, the initiator will be in supervisor state.

- Entry: IEFSD589
- Exit: LINK to IEFSD534, XCTL to IEFSD510 or IEFPPGM
- Tables/Work Areas: Same as caller
- Attributes: Read-only, reenterable
- Control Section: IEFSD589

IEFSD597: Initiator -- Shared DASD ENQ/DEQ Purge Routine

This routine is the purge routine for systems that include the shared DASD feature. In addition to purging all resources enqueued by a job step, but not dequeued, IEFSD597 also releases reserved devices.

- Entry: IEFSD598
- Exit: Return to caller
- Tables/Work Areas: Major QCB, Minor QCB, QEL, TCB, SVRB, CVT, ABTERM
- Attributes: Read-only, reenterable, disabled
- Control Section: IEFSD598

IEFSD598: Initiator -- ENQ/DEQ Purge Routine

This routine purges all resources enqueued by a job step, but not dequeued.

- Entry: IEFSD598
- Exit: Return to caller
- Tables/Work Areas: Major QCB, Minor QCB, QEL, TCB, SVRB, CVT, ABTERM
- Attributes: Read-only, Reenterable, disabled

- Control Section: IEFSD598

IEFSD599: Initiator -- Small Partition Module

This routine provides an interface with the scheduler in a large partition to initiate and terminate small partitions.

- Entry: IEFSD599,SMALLGO
- Exits: ABEND, or XCTL to IEF589SP or IEFSD586
- Tables/Work Areas: SPIL, allocate parameter list (APL), DSOCB, PIB
- Attributes: Read-only, reenterable
- Control Section: IEFSD599
- External Reference: IEFQMUNC

IEFSMFAT: Initiator -- TCTIOT Construction Routine

This routine constructs the TCTIOT, appends it to the TCT, initializes the TCT storage map, and stores the user routine address in the TCT.

- Entry: IEFSMFAT
- Exit: Return to caller
- Tables/Work Areas: PQE, SMCA, TCB, TCT, TCTIOT, TIOT
- Attributes: Reentrant
- Control Sections: IEFSMFAT

IEFSMFIE: Initiator -- User Exit Initialization Routine

This routine initializes the parameter lists for the Job Initiation and Step Initiation user exits.

- Entry: IEFSMFIE
- Exit: Return to caller
- Tables/Work Areas: JCT, JMR, LCT, SCT, TCT
- Attributes: Reentrant
- Control Sections: IEFSMFIE

IEFSMFLK: Termination Routine -- User Exit Initialization Routine

This routine initializes the parameter lists for the Job Termination and Step Termination user exits.

- Entry: IEFACTLK
- Exit: Return to caller
- Tables/Work Areas: JCT, JMR, LCT, SCT, SMCA, TCB, TCT
- Attributes: Reentrant
- Control Sections: IEFACTLK

IEFSMFWI: Termination Routine -- SMF Writer Interface Routine

This routine constructs the SMF job termination and step termination records.

- Entry: IEFSMFWI
- Exit: Return to caller
- Tables/Work Areas: JCT, JMR, LCT, SCT
- Attributes: Reentrant
- Control Sections: IEFSMFWI

IEFUJI: Initiator -- Dummy User Job Initiation Exit Routine

This routine simulates the presence of a user-supplied job initiation exit routine.

- Entry: IEFUJI
- Exit: Return to caller
- Attributes: Reentrant
- Control Sections: IEFUJI

IEFUJV: Interpreter -- Dummy User JCL Validation Exit Routine

This routine simulates the presence of a user-supplied JCL validiation routine.

- Entry: IEFUJV
- Exit: Return to caller
- Attributes: Reentrant
- Control Sections: IEFUJV

IEFUSI: Initiator -- Dummy User Step Initiation Exit Routine

This routine simulates the presence of a user-supplied step initiation exit routine.

- Entry: IEFUSI
- Exit: Return to caller
- Attributes: Reentrant

- Control Sections: IEFUSI

IEFVDA: Interpreter -- DD Statement Processor

This routine constructs and adds entries to a JFCB and SIOT from the complete logical DD statement in the internal text buffer.

- Entry: IEFVDA
- Exit: To IEFVHF
- Tables/Work Areas: IWA, LWA, SIOT, JFCB, JCB, SCT
- Attributes: Read-only, reenterable
- Control Section: IEFVDA

IEFVDBSD: Interpreter -- Data Set Name Table Construction Routine

This routine creates a data set name table.

- Entry: IEFVDBSD
- Exit: To IEFVDA
- Attributes: Reenterable
- Control Section: IEFVDBSD

IEFVEA: Interpreter -- EXEC Statement Processor

This routine constructs or updates an SCT, and, if necessary, a joblib JFCB and SIOT from the complete logical EXEC statement in the internal text buffer.

- Entry: IEFVEA, from IEFVFA
- Exit: To IEFVHF, IEFVINB
- Tables/Work Areas: IWA, EXEC work area, interpreter key table, JCT, SCT, SIOT, QMPA, procedure override table, DCBD, PARMLIST, WORKAREA
- Attributes: Read-only, reenterable
- Control Section: IEFVEA

IEFVFA: Interpreter -- Scan Routine

This routine scans the card image of a JOB, EXEC, or DD statement, performs error checking of JCL syntax, builds internal text, and, when a complete logical statement (including continuations and overrides) has been scanned, passes control to the appropriate statement processor.

- Entry: IEFVFA
- Exits: To IEFVGM, IEFVHQ, IEFVHF, IEFVJA, IEFVDA, IEFVEA
- Tables/Work Areas: IWA, scan routine work area, interpreter key table, QMPA, internal text buffer, scan dictionary.
- Attributes: Read-only, reenterable
- Control Section: IEFVFA

IEFVFB: Interpreter -- Symbolic Parameter Processing Routine

This routine processes symbolic parameters by creating symbolic parameter table buffer entries to assign values to symbolic parameters, and extracts those values and places them in the intermediate text buffer when a symbolic parameter is used.

- Entry: IEFVFB
- Exit: Return to caller
- Tables/Work Areas: IWA, LWA SYMBUF, Intermediate Text Buffer, QMPA
- Attributes: Read-only, reenterable
- Control Section: IEFVFB

IEFVGI: Interpreter -- Dictionary Entry Routine

This routine constructs entries for the refer-back dictionary.

- Entry: IEFVGI
- Exit: Return to caller
- Tables/Work Areas: Refer-back dictionary, auxiliary work area, IWA, QMPA
- Control Section: IEFVGI

IEFVGK: Interpreter -- Get Parameter Routine

This routine searches the internal text buffer for the next parameter, performs basic error checking, and passes control to the appropriate keyword routine.

- Entry: IEFVGK
- Exit: Return to caller
- Tables/Work Areas: Local work area, IWA, internal text buffer, KBT, PDT.
- Control Section: IEFVGK

IEFVGM: Interpreter -- Message Processing Routine

This routine constructs SMBs containing interpreter error messages and JCL statement images, assigns space for these

SMBs in the message class output queue entry, and writes the SMBs into the entry.

- Entry: IEFVGM
- Exit: Return to caller
- Tables/Work Areas: QMPA, SMB, SCD, IWA, JCT
- Attributes: Reenterable, character dependence type C
- Control Section: IEFVGM

IEFVGM1: Interpreter -- Message Module

This routine contains interpreter messages 01-07.

- Attributes: Non-executable
- Control Section: IEFVGM1

IEFVGM2: Interpreter -- Message Module

This routine contains interpreter messages 08-0F.

- Attributes: Non-executable
- Control Section: IEFVGM2

IEFVGM3: Interpreter -- Message Module

This routine contains interpreter messages 10-17.

- Attributes: Non-executable
- Control Section: IEFVGM3

IEFVGM4: Interpreter -- Message Module

This routine contains interpreter messages 18-1F.

- Attributes: Non-executable
- Control Section: IEFVGM4

IEFVGM5: Interpreter -- Message Module

This routine contains interpreter messages 20-27.

- Attributes: Non-executable
- Control Section: IEFVGM5

IEFVGM6: Interpreter -- Message Module

This routine contains interpreter messages 28-2F.

- Attributes: Non-executable
- Control Section: IEFVGM6

IEFVGM7: Interpreter -- Message Module

This routine contains interpreter messages 30-37.

- Attributes: Non-executable
- Control Section: IEFVGM7

IEFVGM8: Interpreter -- Message Module

This routine contains interpreter messages 50-57.

- Attributes: Non-executable
- Control Section: IEFVGM8

IEFVGM9: Interpreter -- Message Module

This routine contains interpreter messages 58-5F.

- Attributes: Non-executable
- Control Section: IEFVGM9

IEFVGM10: Interpreter -- Message Module

This routine contains interpreter messages 60-67.

- Attributes: Non-executable
- Control Section: IEFVGM10

IEFVGM11: Interpreter -- Message Module

This routine contains interpreter messages 68-6F.

- Attributes: Non-executable
- Control Section: IEFVGM11

IEFVGM12: Interpreter -- Message Module

This routine contains interpreter messages 70-77.

- Attributes: Non-executable
- Control Section: IEFVGM12

IEFVGM13: Interpreter -- Message Module

This routine contains interpreter messages 78-7F.

- Attributes: Non-executable
- Control Section: IEFVGM13

IEFVGM14: Interpreter -- Message Module

This routine contains interpreter messages 88-8F.

- Attributes: Non-executable
- Control Section: IEFVGM14

IEFVGM15: Interpreter Message -- Module

This routine contains interpreter messages 90-97.

- Attributes: Non-executable
- Control Section: IEFVGM15

IEFVGM16: Interpreter -- Message Module

This routine contains interpreter messages A0-A7.

- Attributes: Non-executable
- Control Section: IEFVGM16

IEFVGM17: Interpreter -- Message Module

This routine contains interpreter messages 56-5D.

- Attributes: Non-executable
- Control Section: IEFVGM17

IEFVGM18: Interpreter -- Message Module

This routine contains interpreter messages 80-87.

- Attributes: Non-executable
- Control Section: IEFVGM18

IEFVGM19: Interpreter -- Message Module

This routine contains interpreter messages 3E-45.

- Attributes: Non-executable
- Control Section: IEFVGM19

IEFVGM70: Interpreter -- Message Module

This routine contains interpreter messages 38-3F.

- Attributes: Non-executable
- Control Section: IEFVGM70

IEFVGM71: Interpreter -- Message Module

This routine contains interpreter messages 40-47.

- Attributes: Non-executable
- Control Section: IEFVGM71

IEFVGM78: Interpreter -- Message Module

This routine contains interpreter messages 08-0D.

- Attributes: Non-executable
- Control Section: IEFVGM78

IEFVGS: Interpreter -- Dictionary Search Routine

This routine searches the refer-back dictionary for the address of a previously-defined SCT, SIOT, or JFCB.

- Entry: IEFVGS
- Exit: Return to caller
- Tables/Work Areas: Auxiliary work area, IWA, QMPA, refer-back dictionary
- Control Section: IEFVGS

IEFVGT: Interpreter -- Test and Store Routine

This routine performs operations on a parameter as indicated in the appropriate parameter descriptor table entry.

- Entry: IEFVGT
- Exit: Return to keyword routine
- Tables/Work Areas: Internal text buffer, PDT, local work area, IWA
- Control Section: IEFVGT

IEFVHA: Interpreter -- Get Routine

This routine reads statements from the input stream and the procedure library.

- Entry: IEFVHA
- Exits: IEFVHC, IEFVHB, IEFVHAA, IEFSD536, IEFVGM
- Tables/Work Areas: IWA, JCT, DCB
- Attributes: Read-only, reenterable
- Control Section: IEFVHA

IEFVHAA: Interpreter -- End-of-File Routine

This routine determines the conditions under which an end-of-file condition has occurred, and sets switches and passes control accordingly.

- Entry: IEFVHAA
- Exits: IEFVHC or IEFVHN
- Tables/Work Areas: IWA, JCT
- Attributes: Read-only, reenterable
- Control Section: IEFVHAA

IEFVHB: Interpreter -- DD * Statement Generator Routine

This routine generates a "SYSIN DD *" statement for data in the input stream, when no such statement was included.

- Entry: IEFVHB
- Exits: To IEFVHC, IEFVHA, IEFVGM
- Tables/Work Areas: IWA, JCT
- Attributes: Read-only, reenterable
- Control Section: IEFVHB

IEFVHC: Interpreter -- Continuation Statement Routine

This routine determines whether the current statement should be a continuation, and, if so, determines whether it is a valid continuation statement.

- Entry: IEFVHC
- Exits: To IEFVHEB, IEFVHCB, IEFVGM
- Tables/Work Areas: IWA, JCT, DCBD
- Attributes: Read-only, reenterable
- Control Section: IEFVHC

IEFVHCB: Interpreter -- Verb Identification Routine

This routine identifies the verb in a control statement.

- Entry: IEFVHCB
- Exits: To IEFVHE, IEFVHM, IEFVHA, IEFVGM, IEFVHL
- Tables/Work Areas: IWA, JCT, DCBD, PARMLIST, WORKAREA
- Attributes: Read-only, reenterable
- Control Section: IEFVHCB

IEFVHE: Interpreter -- Router

This routine determines the conditions under which it was entered, and passes control to the appropriate routine.

- Entry: IEFVHE
- Exits: To IEFVHEB, IEFVHH, IEFVHEC
- Tables/Work Areas: IWA
- Attributes: Read-only, reenterable
- Control Section: IEFVHE

IEFVHEB: Interpreter -- Pre-Scan Preparation Routine

This routine determines whether a message is required or additional work queue space is required before a statement is scanned. If so, it causes the message to be written or the work queue space to be assigned.

- Entry: IEFVHEB
- Exits: To IEFVHQ, IEFVGM, IEFVHG, IEFVFA
- Tables/Work Areas: IWA, JCT, SCT, QMPA
- Attributes: Read-only, reenterable
- Control Section: IEFVHEB

IEFVHEC: Interpreter -- Job Validity Check Routine

This routine determines whether an SCT has been built for the current job; if not, the routine constructs an SCT.

- Entry: IEFVHEC
- Exits: To IEFVGM, IEFVHH
- Tables/Work Areas: IWA, JCT, SCT
- Attributes: Read-only, reenterable
- Control Section: IEFVHEC

IEFVHF: Interpreter -- Post-Scan Routine

This routine determines the conditions under which it was entered, and passes control accordingly.

- Entry: IEFVHF
- Exits: To IEFVHG, IEFVHEB, IEFVHCB, IEFVHA
- Tables/Work Areas: IWA, CWA
- Attributes: Read-only, reenterable
- Control Section: IEFVHF

IEFVHG: Interpreter -- CPO Routine

This routine writes system input data sets on a direct-access device. If IEFVHG is

unable to obtain enough space to complete writing a data set, control passes to IEFVHR. If the input reaches end-of-file, control passes to IEFVHAA. If a /* is found following DD DATA, control passes to IEFVHA to read the next record. If a // is found, control passes to IEFVHC to identify the verb.

- Entry: IEFVHG
- Exits: To IEFSD536, IEFVGM, IEFVHQ, IEFVHAA, IEFVHA, IEFVHC, or IEFVHB
- Tables/Work Areas: IWA, JCT, SIOT, VOLT, TIOT, LWA, SCT, JFCB, UCB, QMPA, CWA
- Attributes: Read-only, reenterable
- Control Section: IEFVHG

IEFVHH: Interpreter -- Job and Step Enqueue Routine

This routine places the SCT, DSNT, VOLT, and JCT in the job's queue entry, and determines whether the interpreter is to enqueue jobs.

- Entry: IEFVHH
- Exits: To IEFKG, IEFVHQ, IEFSD532, IEFVHHB, IEFVHN
- Tables/Work Areas: IWA, JCT, SCT, QMPA, NEL
- Attributes: Read-only, reenterable
- Control Section: IEFVHH

IEFVHHB: Interpreter -- Housekeeping Routine

This routine initializes for merging a cataloged procedure.

- Entry: IEFVHHB
- Exits: IEFVHA, IEFVHEB
- Tables/Work Areas: IWA
- Attributes: Read-only, reenterable
- Control Section: IEFVHHB

IEFVHL: Interpreter -- Null Statement Routine

This routine determines the conditions under which the null statement was encountered, and passes control to the proper routine.

- Entry: IEFVHL
- Exits: To IEFVHCB, IEFHEC, IEFVHE, IEFVHA
- Tables/Work Areas: IWA, JCT
- Attributes: Read-only, reenterable
- Control Section: IEFVHL

IEFVHM: Interpreter -- Command Statement Routine

This routine tests for valid command verbs, and, if the verb is valid, issues SVC 34 to schedule execution of the command.

- Entry: IEFVHM
- Exits: To IEFVHA, IEFVGM
- Tables/Work Areas: IWA, JCT
- Attributes: Read-only, reenterable
- Control Section: IEFVHM

IEFVHN: Interpreter -- Termination Routine

This routine closes the input stream and procedure library data sets, frees main storage used by the interpreter, and builds the interpreter exit list.

- Entry: IEFVHN
- Exit: Return to caller
- Tables/Work Areas: IWA, JCT, CSCB, QMPA
- Attributes: Read-only, reenterable
- Control Section: IEFVHN

IEFVHQ: Interpreter -- Queue Management Interface Routine

This routine is a common interface between the queue management routines and the interpreter. If an I/O error occurs, IEFVHR receives control. Queue management may be unable to allocate space for a job's input data. If, in this case, the operator replies CANCEL to the message which is issued, IEFVHG receives control.

- Entry: IEFVHQ
- Exits: Return to caller, IEFSD536, or IEFVHG
- Tables/Work Areas: IWA, JCT, QMPA, CSCB
- Attributes: Read-only, reenterable
- Control Section: IEFVHQ

IEFVH1: Interpreter -- Initialization Routine

This routine initializes the interpreter; it obtains main storage for and initializes the IWA, local work areas, and DCBs.

- Entry: IEFVH1
- Exit: To IEFVH2
- Tables/Work Areas: UCB, CSCB, IWA, DCB, local work area
- Attributes: Not reusable
- Control Section: IEFVH1

IEFVH2: Interpreter -- Initialization Routine

This routine opens the input stream data set and the procedure library data set, and obtains main storage for a buffer for procedure library records.

- Entry: IEFVH2
- Exit: To IEFVHA
- Tables/Work Areas: IWA, UCB, TIOT
- Control Section: IEFVH2
- Attributes: Not reusable

IEFVINA: Interpreter -- In-stream Procedure Processor

This routine processes the in-stream procedure. It uses the other in-stream procedure routines as subroutines to perform additional processing.

- Entry: IEFVINA
- Exit: IEFVHA, IEFVHCB
- Tables/Work Areas: EWA, IWA, JCT, PARMLIST, QMPA, WORKAREA
- Attributes: Reenterable
- Control Section: IEFVINA

IEFVINB: In-stream Procedure Search Routine

This routine searches the in-stream directory for a procedure.

- Entry: IEFVINB
- Exit: Return to caller
- Tables/Work Areas: IWA, PARMLIST, QMPA, WORKAREA
- Attributes: Reenterable
- Control Section: IEFVINB

IEFVINC: In-stream Procedure Directory Build Routine

This routine builds a directory entry, if one is needed, for an in-stream procedure.

- Entry: IEFVINC
- Exit: Return to caller
- Tables/Work Areas: IWA, PARMLIST, QMPA, WORKAREA
- Attributes: Reenterable
- Control Section: IEFVINC

IEFVIND: In-stream Procedure Expand Interface Routine

This routine reads a record from the job queue and issues a LOAD macro instruction specifying the expand routine IEZDCODE to expand the record.

- Entry: IEFVIND
- Exit: Return to caller
- Tables/Work Areas: DCBD, IWA, PARMLIST, QMPA
- Attributes: Reenterable
- Control Section: IEFVIND

IEFVINE: In-stream Procedure Syntax Check Routine

This routine syntax checks the PROC and END statements for invalid or non-existant labels and/or null operands with comments.

- Entry: IEFVINE
- Exit: Return to caller
- Tables/Work Areas: 256 byte translate and test table
- Attributes: Reenterable
- Control Section: IEFVINE

IEFVJA: Interpreter -- Job Statement Processor

This routine initializes a JCT and job ACT from the complete logical job statement in the internal text buffer.

- Entry: IEFVJA

- Exit: To IEFVHF
- Tables/Work Areas: IWA, job work area, interpreter key table, JCT, ACT, QMPA
- Attributes: Read-only, reenterable
- Control Section: IEFVJA

IEFVJIMP: Termination -- JOB Statement Condition Code Processor

This routine tests the condition codes specified in the JOB statement to determine whether the remaining steps in the job are to be run.

- Entry: IEFVJ
- Exits: To IEFVK or IEFZA
- Tables/Work Areas: LCT, JCT, SCT
- Control Section: IEFVJ

IEFVJMSG: Termination -- JOB Statement Condition Code Processor Messages

This routine contains the messages issued to the programmer by the JOB statement condition code processor.

- Entry: IEFVJMSG
- Attributes: Non-executable
- Control Section: IEFVJMSG

IEFVKIMP: I/O Device Allocation -- EXEC Statement Condition Code Processor

This routine tests the condition codes specified in the EXEC statement to determine whether the next step of the job is to be run.

- Entry: IEFVK
- Exits: IEFVS, IEFLB
- Tables/Work Areas: JCT, LCT, SCT
- Control Section: IEFVK

IEFVKMSG: I/O Device Allocation -- EXEC Statement Condition Code Processor Messages

This routine contains the messages issued to the programmer by the EXEC -- statement condition -- code processor.

- Entry: IEFVKMJ1
- Attributes: Non-executable
- Control Section: IEFVKMSG

IEFVMFAK: I/O Device Allocation -- Linkage to IEFVMLS1

This routine passes control to entry point IEFVMCVL of the JFCB housekeeping module IEFVMLS1 via the XCTL macro instruction.

- Entry: IEFVMCVL
- Exit: To IEFVMCVL
- Control Section: IEFVMCVL

IEFVMLS1: I/O Device Allocation -- JFCB Housekeeping Control Routine and Allocate Processing Routines

The control routine obtains the required SIOTs, determines the processing required for each, and passes control to the appropriate routine. The allocate processing routine performs the processing required in certain refer-back situations, when the data set is cataloged or passed, and when unit name is specified.

- Entry: IEFVM, IEFVMCVL, IEFVMQMI, VM7000, VM7055, VM7055AA, VM7060, VM7070, VM7090, VM7130, VM7370, VM7700, VM7742, VM7750, VM7850, VM7900, VM7950
- Exits: To IEFVM2LS, IEFVM3LS, IEFVM4LS, IEFVM5LS, IEFVM6LS, and IEFXCSSS, IEFDSOAL
- Tables/Work Areas: LCT, JCT, PDQ, SIOT, JFCB, QMPA
- Control Section: IEFVM1

IEFVMLS6: I/O Device Allocation -- JFCB Housekeeping Error Message Processing Routine

This routine prepares error messages for the JFCB housekeeping routines.

- Entry: IEFVMSGR
- Exit: Return to caller
- Tables/Work Areas: JCT, LCT
- Control Section: IEFVM6

IEFVMLS7: I/O Device Allocation -- JFCB Housekeeping Error Messages

This routine contains the messages issued by the JFCB housekeeping routines.

- Entry: IEFVM7
- Attributes: Non-executable
- Control Section: IEFVM7

IEFVMMS1: I/O Device Allocation -- Linkage to JFCB Housekeeping

This routine provides a linkage to the JFCB housekeeping routines for the step flush function.

- Entry: IEFVM1
- Exit: XCTL to IEFVMLS1
- Attributes: Read-only, reenterable
- Control Section: IEFVM1

IEFVM2LS: I/O Device Allocation -- JFCB Housekeeping Fetch DCB Processing Routine

This routine updates the SIOT, SCT, JFCB and VOLT with information required for the allocation of devices for the fetch DCB.

- Entry: VM7100
- Exit: To IEFVMLS1
- Tables/Work Areas: LCT, SCT, SIOT, JFCB, VOLT
- Control Section: IEFVM2

IEFVM3LS: I/O Device Allocation -- JFCB Housekeeping GDG Single Processing Routine

This routine obtains the fully qualified name of a member of a generation data group (GDG), and completes the required information in the JFCB, VOLT, and SIOT for that member.

- Entry: VM7150
- Exit: To IEFVMLS1
- Tables/Work Areas: LCT, SIOT, GDG Bias Count table, JFCB
- Control Section: IEFVM3

IEFVM4LS: I/O Device Allocation -- JFCB Housekeeping GDG All Processing Routine

This routine builds an SIOT, JFCB, and VOLT, and PDQ entries for each member of the GDG.

- Entry: VM7200
- Exit: To IEFVMLS1
- Tables/Work Areas: LCT, SCT, VOLT, PDQ, SIOT, JFCB
- Control Section: IEFVM4

IEFVM5LS: I/O Device Allocation -- JFCB Housekeeping Patterning DSCB Routine

This routine establishes DCB control information within a JFCB.

- Entry: VM7300
- Exit: To IEFVMLS1
- Tables/Work Areas: LCT, SCT, SIOT, DSCB, JFCB
- Control Section: IEFVM5

IEFVM76: I/O Device Allocation -- JFCB Housekeeping Unique Volume ID Routine

This routine creates unique volume serials for unlabeled tape data sets, when the disposition is "PASS".

- Entry: VM7600
- Exit: Return to caller
- Tables/Work Areas: SIOT, JFCB, JFCBX
- Control Section: IEFVM76

IEFVRRC: Reinterpretation Control Routine

This routine passes control among the routines that modify the queue entry of a restart step so that they appear as they were prior to the initiation of the step.

- Entry: IEFVRRC, IEFVRRCA, IEFVRRCB
- Exit: Return to caller
- Attributes: Read-only reenterable
- Tables/Work Areas: NEL, JCT, SCT, SIOT, JFCB, JFCBX, VOLT, SMB, DSENQ, SCD, DSB, QMPA
- Control Section: IEFVRRC

IEFVRR1: Dequeue Interface Routine

This routine interfaces with queue management to cause a specific job to be dequeued and the JCT for that job to be read into main storage.

- Entry: IEFVRR1
- Exit: Return to caller
- Tables/Work Areas: QMPA, JCT
- Attributes: Read-only, reenterable
- Control Section: IEFVRR1

IEFVRR2: Table Merge Routine

This routine merges the reinterpreted queue entry tables of a restart step with the original queue tables for that step.

- Entry: IEFVRR2, IEFVR2AE
- Exit: Return to caller
- Tables/Work Areas: QMPA, JCT, ACT, SMB, SCT, SIOT, JFCB, DSENQ, VOLT, JFCBX, NEL
- Attributes: Reenterable
- Control Section: IEFVRR2

IEFVRR3: Reinterpretation Delete/Enqueue Routine

This routine deletes the reinterpreted input and output queue entries of a restart step, constructs the internal JCL necessary for processing a checkpoint restart, and reenqueues the job's queue entry.

- Entry: IEFVRR3, IEFVR3AE
- Exit: Return to caller
- Tables/Work Areas: QMPA, JCT, SCT, SIOT, JFCB
- Attributes: Reenterable
- Control Section: IEFVRR3

IEFVSDRA: Restart Activation Routine

This routine issues a START Restart Reader command for one or more jobnames. This routine is entered from IEFSD168 during a problem program restart or IEFSD305 during a warm start.

- Entry: IEFVSDRA
- Exit: Return to caller
- Tables/Work Areas: CSCB, CVT, TCB
- Attributes: Reenterable
- Control Section: IEFVSDRA

IEFVSDRD: Restart Determination Routine

This routine initiates automatic restarts.

- Entry: IEFVSDRD
- Exit: To IEFSD305
- Tables/Work Areas: JCT, SCT, QMPA, CVT, TIOT, LCT
- Attributes: Reenterable
- Control Section: IEFVSDRD

IEFVSD12: Interpreter -- CPO Allocation Subroutine

This routine sets up a JFCB and allocates space on a direct-access device for a system input data set.

- Entry: IEFSD012
- Exit: Return to caller
- Attributes: Reenterable
- Tables/Work Areas: IWA, QMPA, LWA, SIOT, TIOT, UCB, JFCB, JCT, CSCB
- Control Section: IEFSD012
- External References: IEFVHQ

IEFVSD13: Interpreter -- SCD Construction Routine

This routine constructs an SCD entry for each system output class defined for a job, and assigns space for all DSBs that will be required.

- Entry: IEFSD090
- Exit: Return to caller
- Tables/Work Areas: IWA, QMPA, DD work area, SCD, SCT, SIOT, JCT, JFCB
- Control Section: IEFSD090

IEFVSMBR: SMB Reader Routine

This routine reads the SMBs associated with a restarting job and converts the JCL statements to their original format.

- Entry: IGC0005B
- Exits: If called during restart reader processing, return to caller; if called during restart, XCTL to the first load of restart housekeeping.
- Tables/Work Areas: QMPA, DCB, JCT, SMB, RRCWKAR, SCT
- Attributes: Reenterable
- Control Section: IEFVSMBR

IEFWA000: I/O Device Allocation -- Demand Allocation Routine

This routine establishes data set device requirements, and allocates in response to specific unit requests.

- Entry: IEFWA000, IEFUCBL
- Exits: To IEFWD000, IEFX3000, IEFX5000
- Tables/Work Areas: UCB Address List, DMT, UCB, LCT, SCT, SIOT, VOLT, AWT
- Control Sections: IEFWA7, IEFWA002

IEFWCFAK: I/O Device Allocation -- Linkage Module

This routine passes control to the TIOT construction routine.

- Entry: IEFWC000, IEFWC002
- Exit: To IEFWCIMP
- Control Section: IEFWC000, IEFWC002

IEFWCIMP: I/O Device Allocation -- TIOT Construction Routine

This routine calculates the main storage required for the TIOT, builds the TIOT, and processes requests for direct-access space.

- Entry: IEFWC000
- Exits: To IEFXJIMP, IEFWD000
- Tables/Work Areas: JCT, SCT, LCT, SIOT, VOLT, AWT, TIOT
- Control Section: IEFWC000

IEFWDFAK: I/O Device Allocation -- Linkage Module

This routine passes control to the external action routine.

- Entry: IEFWD000
- Exit: To IEFWD000
- Control Section: IEFWD000

IEFWD000: I/O Device Allocation -- External Action Routine

This routine ensures that the correct volumes are mounted on the appropriate units for new data sets and that SCRATCH volumes are mounted on the appropriate devices. It also sets up interfaces for the mounting of volumes needed for old data sets.

- Entry: IEFWD000, IEFWDMSG
- Exits: To IEFXT000, IEFW41SD, IEFXK000
- Tables/Work Areas: SCT, LCT, TIOT, UCB, JFCB, ATCA, MVA, MVAX
- Control Section: IEFWD000, IEFWDMSG, IEFWD002

IEFWD001: I/O Device Allocation -- External Action Messages

This routine contains a directory and the messages used in the external action routine.

- Entry: IEFWD001
- Attributes: Non-executable
- Control Section: IEFWD001

IEFWEXTA: I/O Device Allocation -- Extended External Action Routine

This routine ensures that the correct volumes are mounted on the appropriate units for old data sets.

- Entry: IEFWEXTA
- Exits: to IEFW41SD, IEFXK000
- Tables/Work Areas: CSCB, LCT, MVI, SCT, UCB
- Control Section: IEFWEXTA

IEFWSMSG: Termination-Message Routine for Warmstart

This routine contains messages to be used by module IEFWSYP3 during warmstart.

- Entry: IEFWSMSG
- Attributes: Non-executable
- Control Section: IEFWSMSG

IEFWSTRT: I/O Device Allocation -- Message Module

This routine contains the message issued to the operator when a job is started and the messages issued to the operator when a job is terminated due to ABEND, condition codes, or JCL errors.

- Entry: IEFWSTRT
- Attributes: Non-executable
- Control Section: IEFWSTRT

IEFWSWIN: I/O Device Allocation -- Linkage Module

This routine passes control to the decision allocation routine.

- Entry: IEFWSWIT

- Exit: To IEFX5000
- Control Section: IEFWSWIT

IEFWSYP3: Termination-SIOT Reader Routine for Warmstart

This routine either matches TIOT entries with SIOTs or, in the case of dynamically allocated data sets, builds TIOT entries for use by the termination routines.

- Entry: IEFWSYP3
- Exit: Return to caller
- Tables/Work Areas: QMPA, JCT, LCT, SIOT, UCB, TIOT, SIOTTTR, TIOTEXT
- Control Section: IEFWSYP3

IEFWTERM: Termination -- Message Module

This routine contains the message issued to the operator when a job is terminated normally, or when it is terminated because of a JCL error found in the interpreter or initiator.

- Entry: IEFWTERM
- Attributes: Non-executable
- Control Section: IEFWTERM

IEFWTP00: Write-to-programmer Initialization Routine

This routine initializes storage and registers to process write-to-programmer messages if the WTP call is valid.

- Entry: IGC0203E from IEECVWTO
- Exits: Normal to IEFWTP01, to calling program if only a WTP is requested and the WTP limit has been exceeded, to IEECVWTO if WTP request cannot be processed or a WTO message was also requested.
- Tables/Work Areas: WTPCB, JSCB, UCM, CVT, WPL, IEFQMNGR, IEFQMRES
- Attributes: Reenterable
- Control Section: IGC0203E

IEFWTP01: Write-to-programmer Message Processing Routine

This routine processes the WTP messages and writes them on the job queue using the transient queue manager (SVC-90).

- Entry: IGC0303E from IEFWTP00 or IEFWTP02
- Exits: Normal to IEECVWTO or to calling program if only a WTP is requested; to IEFWTP02 for processing I/O errors which occur while writing a WTP message or for job queue problems.
- Tables/Work Areas: WTPCB, JSCB, UCM, CVT, WPL, IEFQMNGR, IEFQMRES
- Attributes: Reenterable
- Control Section: IGC0303E

IEFWTP02: Write-to-programmer Error Processing Routine

This routine handles WTP processing using the reserved WTP SMBs for messages when there are I/O errors in the job queue or when WTP is unable to get a record assigned for a WTP message using the transient queue manager.

- Entry: IGC0403E from IEFWTP01
- Exits: Return to IEECVWTO, to calling program if only a WTP was requested, or IEFWTP01 if a system WTP message is to be processed following a NO RECORD message.
- Tables/Work Areas: WTPCB, JSCB, UCM, CVT, WPL, IEFQUNGR, IEFQMRES
- Attributes: Reenterable
- Control Section: IGG0403E

IEFXAMSG: I/O Device Allocation -- Message Module

This routine contains the messages issued by the allocation control routine.

- Entry: IEFXAMSG
- Attributes: Non-executable
- Control Section: IEFXAMSG

IEFXCSSS: I/O Device Allocation -- Allocation Control Routine

This routine calculates table space requirements and obtains the main storage for the tables used or built during allocation.

- Entry: IEFXA
- Exits: To IEFXJ, IEFWA, IEFWC
- Tables/Work Areas: JCT, SCT, LCT, UCB, SIOT, VOLT, AWT
- Control Section: IEFXA

IEFXH000: I/O Device Allocation -- Separation Strikeout Routine

This routine strikes from AWT entries, the bits corresponding to devices that would violate separation or affinity requests.

- Entry: IEFXH000
- Exit: Return to caller
- Tables/Work Areas: LCT, AWT, AVT, UCB
- Control Section: IEFXH000

IEFXJFAK: I/O Device Allocation -- Linkage Module

This routine passes control to the allocation recovery routine.

- Entry: IEFXJ000
- Exit: To IEFXJIMP
- Control Section: IEFXJ000

IEFXJIMP: I/O Device Allocation -- Allocation Recovery Routine

This routine informs the operator of the allocation recovery options available, and passes control to the proper routine to comply with his request.

- Entry: IEFXJ000, IEFV15XL, IEFXJX5A
- Exits: To IEFXCSSS, IEFSD095, IEFW41SD
- Tables/Work Areas: LCT, AWT, JCT, CVT, UCB, SCT, SIOT
- Control Section: IEFXJ000

IEFXJMSG: I/O Device Allocation -- Allocation Recovery Messages

This routine contains the messages used by the allocation recovery routine.

- Entry: MSRCV, MSSYS, MSOFF
- Attributes: Non-executable
- Control Section: IEFXJMSG

IEFXKIMP: I/O Device Allocation -- Non-Recovery Error Routine

This routine cancels the step when a lack of available devices has been discovered after the TIOT is constructed.

- Entry: IEFXK000
- Exit: To IEFW41SD
- Tables/Work Areas: LCT, SCT, UCB, TIOT
- Control Section: IEFXK000

IEFXKMSG: I/O Device Allocation -- Non-Recovery Error Routine Messages

This routine contains the messages used by the non-recovery error routine.

- Entry: IEFXKMSG
- Attributes: Non-executable
- Control Section: IEFXKMSG

IEFXQM00: Transient Queue Manager Initialization and Read/Write Routine

This routine initializes tables and read or writes job queue records.

- Entry: IGC00090
- Exits: XCTL to IGC01090 or return to caller
- Tables/Work Areas: Q/M resident data area, QMPA, CVT, ECB/IOB
- Attributes: Reenterable
- Control Section: IGC00090

IEFXQM01: Transient Queue Manager Track Assignment Routine

This routine assigns logical tracks as required.

- Entry: IGC01090
- Exits: XCTL to IGC02090 or return to caller
- Tables/Work Areas: QM resident data area, QMPA, CVT, ICB/IOB
- Attributes: Reenterable
- Control Section: IGC01090

IEFXQM02: Transient Queue Manager Record Assignment Routine

This routine assigns records to a queue entry.

- Entry: IGC02090
- Exits: Return to caller
- Tables/Work Areas: QM resident data area, QMPA, CVT, ECB/IOB
- Attributes: Reenterable

- Control Section: IGC02090

IEFXT00D: I/O Device Allocation -- Space Request Routine

This routine obtains space on direct-access devices for requesting data sets.

- Entry: IEFXT000
- Exits: To IEFW41SD, IEFXK000, IEFWD000
- Tables/Work Areas: LCT, TIOT, UCB, JCT, SIOT, JFCB, PDQ
- Control Section: XTTP00, IEFXT000

IEFXT002: I/O Device Allocation -- VARY Command Interface TIOT Compression Routine

This routine reduces the TIOT to its final size and provides an interface with the VARY command.

- Entry: IEFXT002, XTTRDJ, XTTEB3, XTTEA1, XTTEA0
- Exits: to IEFXKIMP, IEFXT003, IEFWEXTA
- Tables/Work Areas: LCT, TIOT, UCB, JCT, SIOT, JFCB
- Control Section: IEFXT002

IEFXT003: I/O Device Allocation -- DADSM Error Recovery Routine

This routine determines what action should be taken when the request for space on a particular volume fails.

- Entry: IEFXT003, XUUH06, XUUB00
- Exits: To IEFXT00D, IEFXT002
- Tables/Work Areas: LCT, TIOT, UCB, JCT, SIOT, JFCB
- Control Section: IEFXT003

IEFXVMSG: I/O Device Allocation -- Automatic Volume Recognition Messages

This routine contains the messages used by the automatic volume recognition (AVR) routine.

- Entry: IEFXVMSG
- Attributes: Non-executable
- Control Section: IEFXVMSG

IEFXVNSL: I/O Device Allocation -- Automatic Volume Recognition -- Non-Standard Label Routine

This routine processes non-standard labels for the AVR routine.

- Entry: IEFXVNSL
- Exit: Return to caller
- Control Section: IEFXVNSL

IEFXV001: I/O Device Allocation -- Automatic Volume Recognition Routine

This routine finds and allocates volumes pre-mounted by the operator.

- Entry: IEFXV001
- Exits: IEFWC000, IEFX5000, IEFXJ000
- Tables/Work Areas: JCT, SCT, AWT, AVT, VOLT, SIOT, LCT, UCB
- Control Section: IEFXV001

IEFXV002: I/O Device Allocation -- Automatic Volume Recognition, Label Processing

This routine reads the label of a newly mounted volume, extracts the serial number, and places it into the UCB for the corresponding device.

- Entry: IEFXV002
- Exits: To IEFXVNSL via CALL, return to caller.
- Tables/Work Areas: LUT, UCB, CVT, DEB, IOB
- Attributes: Reusable
- Control Section: IEFXV002

IEFXV003: I/O Device Allocation -- AVR Unmounted Tape Allocation Routine

This routine determines the best allocation solution for unmounted 3400 9-trk and 2400 9-trk tape requests based on the device configuration.

- Entry: IEFXV003
- Exit: To IEFXV002, return to caller
- Tables/Work Areas: ACB, AWT, UCB, LCT
- Attributes: Reusable
- Control Section: IEFXV003

IEFX300A: I/O Device Allocation -- Device Strikeout Routine

This routine modifies the primary and secondary bit patterns in AWT entries to complete the allocation to a data set.

- Entry: IEFX3000, X33B42
- Exit: Return to caller
- Tables/Work Areas: AWT, AVT, UCB, LCT
- Control Section: IEFX3000

IEFX5000: I/O Device Allocation -- Decision Allocation Routine

This routine selects devices for data sets with multiple unit possibilities.

- Entry: IEFX5000, XIIB32, X55C86, X55D3G
- Exits: To IEFWC000, IEFXJ000
- Tables/Work Areas: LCT, AWT, AVT, UCB
- Control Section: IEFX5000

IEFYNIMP: Termination -- Step Termination Control Routine

This routine passes control among the modules of the step termination routine and, when required, passes control to the job termination routine.

- Entry: IEFYN
- Exits: To IEFW22SD, IEFYPJB3, IEFVJIMP, IEFZAJB3, IEFRPREP
- Tables/Work Areas: JCT, SCT, LCT
- Control Section: IEFYN

IEFYNMSG: Termination -- Step Termination Control Routine Messages

This routine contains the messages required for the step termination control routine.

- Entry: IEFYNMSG, STRMSG01
- Attributes: Non-executable
- Control Section: IEFYNMSG

IEFYPJB3: Termination -- Step Termination Data Set Driver Routine

This routine obtains SIOTs and to pass control to the disposition and unallocation routine.

- Entry: IEFYP
- Exits: To IEFZG, IEFYNIMP
- Tables/Work Areas: LCT, TIOT, UCB, QMPA, SIOT, TCB
- Control Section: IEFYP

IEFYPMSG: Termination -- Step Termination Messages

This routine contains the messages required by the step termination routine.

- Entry: IEFYPMSG, YPPMSG1, YPPMSG2
- Attributes: Non-executable
- Control Section: IEFYPMSG

IEFYSVMS: Termination -- Message Blocking Routine

This routine blocks system messages into SMBs, and places SMBs into the message class queue entry.

- Entry: IEFYS
- Exit: Return to caller
- Tables/Work Areas: LCT, SCT, SMB
- Attributes: Reenterable
- Control Section: IEFYS

IEFYTVMS: Termination -- DSB Processing Routine

This routine places data set blocks in the space reserved for them in the output queue entries. If a job used DSO, the data set blocks are marked inactive.

- Entry: IEFYT
- Tables/Work Areas: JCT, SCT, TIOT, SIOT, QMPA, DSCB, LCT, CVT, JFCB, DSOCB, PIB
- Attributes: Reenterable
- Control Section: IEFYT

IEFZAJB3: Termination -- Job Termination Control Routine

This routine provides entry to the job termination routine, obtains PDQ blocks, and passes control to the disposition and unallocation routine.

- Entry: IEFZA
- Exits: To IEFZGJ, IEFW31SD
- Tables/Work Areas: LCT, JCT, PDQ, UCB, QMPA

- Control Section: IEFZA

IEFZGJB1: Termination -- Disposition and Deallocation Routine

This routine directs the disposition and deallocation of those data sets that remain to be processed at job termination: passed data sets that were not received, and retained data sets that were not referred to.

- Entry: IEFZGJ, ZP0QM
- Exit: Return to caller
- Tables/Work Areas: JCT, PDQ, JFCB, LCT, QMPA, UCB
- Control Section: IEFZGJ

IEFZGMSG: Termination -- Disposition and Deallocation Messages

This routine contains the messages required for the disposition and deallocation routine (IEFZGJB1).

- Entry: IEFZGMSG
- Attributes: Non-executable
- Control Section: IEFZGMSG

IEFZGST1: Termination -- Disposition and Deallocation Routine

This routine directs the disposition of data sets as specified in the DISP field of the DD statement, and makes the associated units available for allocation to other data sets.

- Entry: IEFZG, ZP0QMGR1
- Exit: Return to caller
- Tables/Work Areas: LCT, PDQ, SIOT, TIOT, UCB, JFCB, QMPA
- Control Section: IEFZG

IEFZGST2: Termination -- Unallocation Routine

This routine makes available to other data sets the units used by the terminating job step.

- Entry: IEFZG2, ZGOK09, ZOOA1, ZOOE10, ZPOC10, ZPOQMGR2
- Exit: Return to caller
- Tables/Work Areas: LCT, PDQ, SIOT, TIOT, UCB, JFCB, QMPA
- Control Section: IEFZG2

IEFZHMSG: Termination -- VARY Command Interface and Disposition and Deallocation Message Routine

This routine prepares messages to the programmer and to the operator for the disposition and allocation routines. It also provides an interface with the VARY command.

- Entry: IEFZH, ZG0E60, ZK0D1, ZK0E1, XPS631
- Exit: Return to caller
- Tables/Work Areas: LCT, QMPA, SMB
- Control Section: IEFZH

IEF078SD: System Output Writer -- Linkage Module

This routine transfers control to module IEFSD078.

- Entry: IEFSD078
- Exit: To IEFSD078
- Attributes: Reenterable

IEF079SD: System Output Writer -- Linkage Module

This routine transfers control to IEFSD079.

- Entry: IEFSD079
- Exit: To IEFSD079
- Attributes: Reenterable

IEF082SD: System Output Writer -- Linkage Module

This routine passes control to the system output writer main processing routine.

- Entry: IEFSD082
- Exit: To IEFSD082
- Control Section: IEFSD082

IEF083SD: System Output Writer -- Linkage Module

This routine passes control to the system output writer command processing routine.

- Entry: IEFSD083
- Exit: IEFSD083
- Control Section: IEFSD083

IEF300SD: System Restart -- Linkage Module

This routine provides a linkage to the system restart initialization routine.

- Entry: IEFSD300
- Exits: To IEFSD300, IEFSD055
- Attributes: Reenterable

IEF304SD: System Restart -- Linkage Module

This routine provides a linkage to the system restart scratch data sets routine.

- Entry: IEFSD304
- Exits: To IEFSD304, IEFSD055
- Attributes: Reenterable
- Control Section: IEFSD304

IEF41DUM: Allocation -- Return to Initiator or System Task Control

This routine returns control to the Initiator or to System Task Control after device allocation has completed. If allocation was called by the Initiator, the routine returns control to step initiation routine IEFSD512 at entry point IEFALRET via an XCTL macro instruction. If allocation was called by System Task Control, the routine returns control to the caller.

- Entry: IEFSD41R
- Exits: IEFALRET or return to caller
- Tables/Work Areas: CSCB, LCT
- Attributes: Reenterable
- Control Section: IEFSD41R

IEF41FAK: I/O Device Allocation -- Linkage Module

This routine provides a linkage to the allocation exit routine during step flush.

- Entry: IEFW41SD, IEFW1FAK, IEFW2FAK
- Exit: To IEFW41SD
- Attributes: Read-only, reenterable
- Control Section: IEFW41SD

IEF589SP: System Task Control -- Linkage to IEFSD584

This routine links to IEFSD584 so that upon return, the initiator will be in supervisor state.

- Entry: IEF589SP from IEFSD599
- Exit: Link to IEFSD584, XCTL to IEFSD599
- Control Section: IEF589SP

IEZDCODE: Interpreter -- In-stream Procedure Expand Routine

This routine expands a given statement to its original form by inserting a given character.

- Entry: IEZDCODE
- Exit: Return to caller
- Tables/Work Areas: IEZPARM
- Attributes: Reenterable
- Control Section: IEZDCODE

IEZNCODE: Interpreter -- In-stream Procedure Compress Routine

This routine compresses a given statement by removing a given character. The new statement that is formed contains the number of the removed character.

- Entry: IEZNCODE
- Exit: Return to caller
- Tables/Work Areas: IEZPARM
- Attributes: Reenterable
- Control Section: IEZNCODE

IGC0008C: SVC 83 -- SMF Buffer Manager and Writer Routine

The purpose of this routine is to move SMF records into the SMF buffer, and to cause the records to be written when the buffer is full.

- Entry: IGC00083
- Exit: IEESMFOP, return to caller
- Tables/Work Areas: CVT, SMCA
- Attributes: Reentrant
- Control Sections: IGC00083

IGC0103D: SVC 34 -- Master Command EXCP Routine

This routine processes the MOUNT Command.

- Entry: IGC0103D

- Attributes: Reenterable, transient
- Control Section: IGC0103D.

IGF08501: SVC 34 -- Machine Status Control Routine - Model 85 (1)

This routine is available only for the model 85. It processes the status parameter of the MODE command.

- Entry: IGF08501
- Exit: IGF08502
- Tables/Work Areas: CVT, XSA, MSB
- Attributes: Reenterable, read-only, self-relocating
- Control Section: IGF08501

IGF08502: SVC34 - Machine Status Control Routine - Model 85 (2)

This routine is available only for the model 85. It processes all parameters of the MODE command but the status parameter.

- Entry: IGF08502
- Exit: Return to issuer of SVC 34
- Tables/Work Areas: CVT, XSA
- Attributes: Reenterable, read-only, self-relocating
- Control Sections: IGF08502

IGF13501: SVC 34 -- Machine Status Control Routine - Model 135

This routine processes the MODE command for the Model 135.

- Entry: IGF13501
- Exit: Return to the issuer of SVC 34
- Tables/Work Areas: CVT, MSB, XSA
- Attributes: Reenterable, read-only, transient
- Control Section: IGF13501

IGF2403D: SVC 34 - VARY PATH Command Processor

This routine is used only if the Alternate Path Retry option is supported. It processes an operator's request for varying a PATH to a device and causing that path to be either logically brought online or logically removed from the system.

- Entry: IGF2403D from IEE3202D
- Exit: To IEE0503D, IEE2103D
- Attributes: Reenterable
- Tables/Work Areas: Test Channel Table, SVRB Extend Save Area
- Control Sections: IGF2403D

IGF2503D: SVC 34 - SWAP Command Processor

This routine is used only if Dynamic Device Reconfiguration is in the system. It processes the operator's command to SWAP volumes for Dynamic Device Reconfiguration. The routine checks the command for proper format, sets a switch if the status of DDR is to be changed, validates CUAs, and fills in certain fields of the I/O RMS Communications Area.

- Entry: IGF2503D from IEE0403D or from IGF0408E (DDR Tape Reposition)
- Exit: To IEE0503D, IEEZ103D, IGF0408E (DDR Tape Reposition)
- Attributes: Reenterable
- Tables/Work Areas: I/O RMS Communications Area, SVRB Extended Save Area
- Control Sections: IGF2503D

IGF2603D: SVC 34 -- MODE Command Router Routine

This routine accepts the MODE command and passes control to the appropriate machine status control routine depending on the machine model.

- Entry: IGF2603D
- Exit: XCTL to IGF05801 (for Model 85), to IGF29601 (for Model 155), or to IGF55301 (for Model 165)
- Tables/Work Areas: MSB, XSA
- Attributes: Reenterable, read-only, self-relocating, transient
- Control Section: IGF2603D

IGF29601: SVC 34 -- Machine Status Control Routine - Model 155

This routine processes the MODE command for the Model 155.

- Entry: IGF29601

- Exit: Return to issuer of SVC 34
- Tables/Work Areas: CVT, MSB, XSA
- Attributes: Serially reusable, read-only, self-relocating
- Control Section: IGF29601

IGF29701: SVC 34 -- Machine Status Control Routine - Model 145

This routine processes the MODE command for the Model 145.

- Entry: IGF29701
- Exit: Return to issuer of SVC 34
- Tables/Work Areas: CVT, MSB, XSA
- Attributes: Reenterable, read-only, transient
- Control Section: IGF29701

IGF55301: SVC 34 -- Machine Status Control Routine - Model 165

This routine processes the MODE command for the Model 165.

- Entry: IGF55301
- Exit: Return to issuer of SVC 34
- Tables/Work Areas: CVT, MSB, XSA
- Attributes: Reenterable, read-only, transient
- Control Section: IGF55301

IKJNULL: SVC 34 -- Command Rejector Routine

This routine is used by MFT and non-TSO MVT systems to reject the DISPLAY USER command, because this command is not relevant in a non-TSO environment.

- Entry: IKJNULL, from IEE3503D
- Exit: IEE2103D
- Tables/Work Areas: XSA
- Attributes: Reentrant, self-relocating, read-only, transient
- Control Section: IKJNULL

APPENDIX C: FLOWCHARTS

This appendix includes the MFT job management flowcharts that are different from MVT. For the flowcharts on allocation, termination, and system restart, see IBM System/360 Operating System: MVT Job Management, Program Logic Manual, GY28-6660.

Chart 1A. Small Partition Routine (Part 1 of 4)

Note -
At Entry, Small Partition Has Zero Protection in TCB, PSW and Hardware. Also, PSW is Supervisor State.

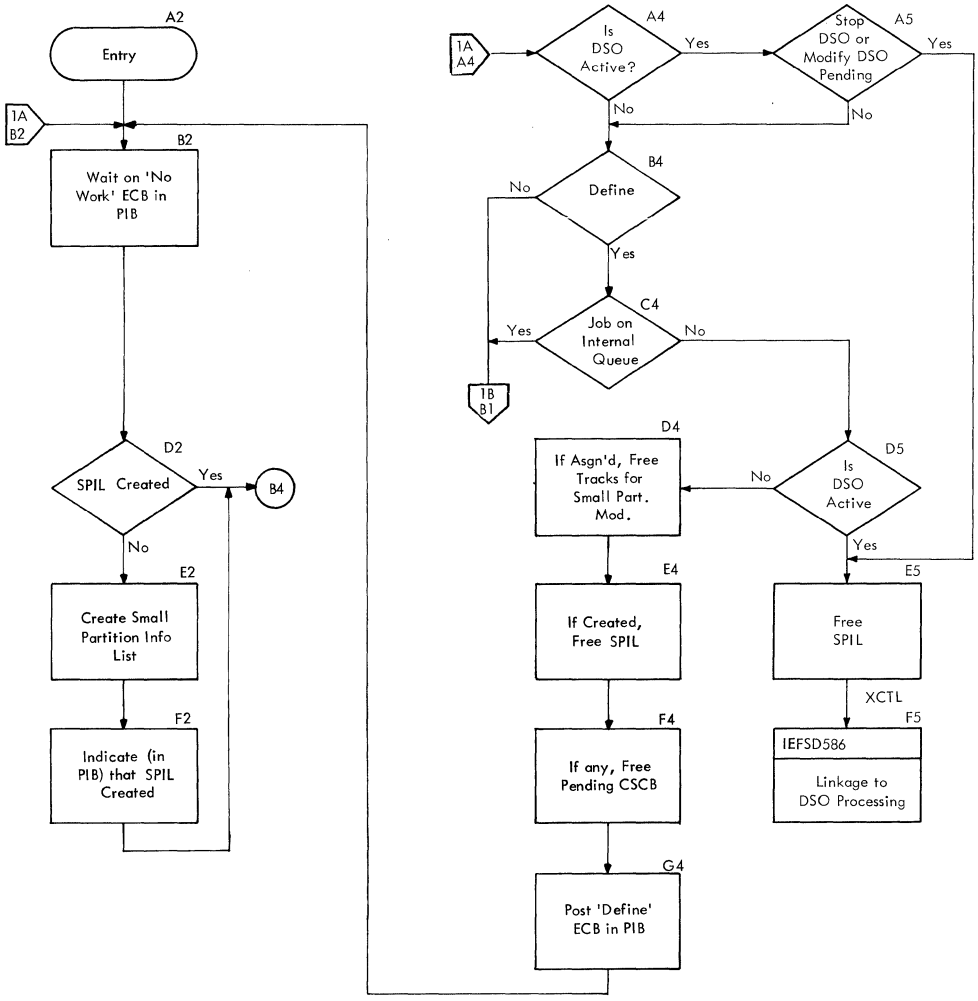


Chart 1B. Small Partition Routine (Part 2 of 4)

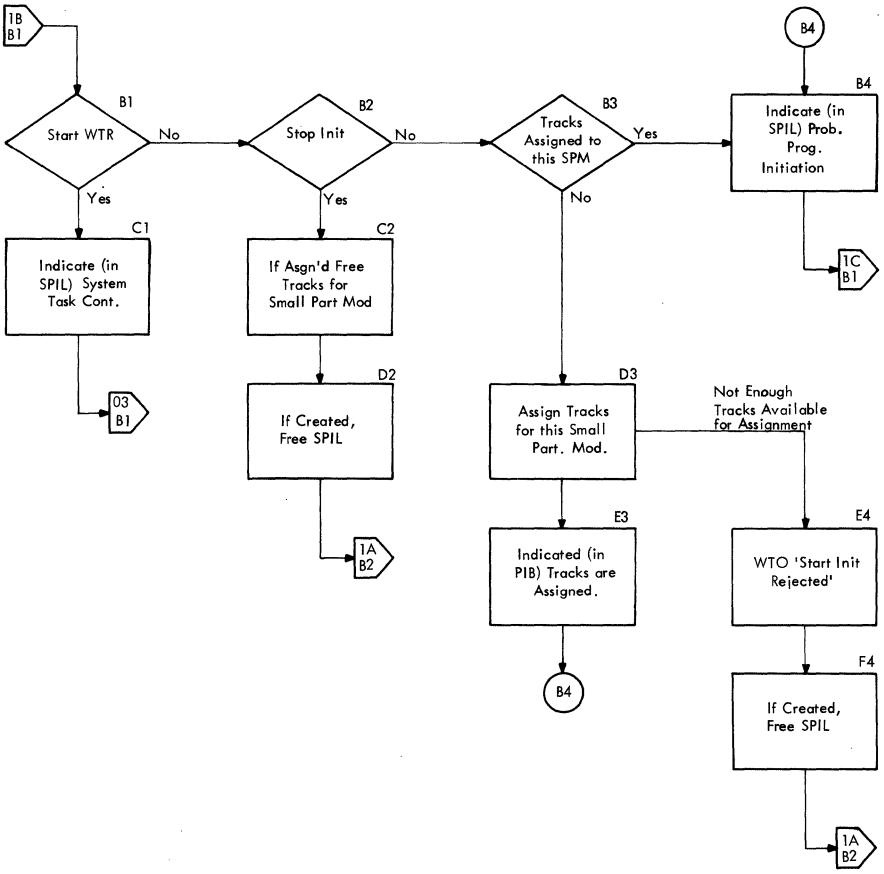


Chart 1C. Small Partition Routine (Part 3 of 4)

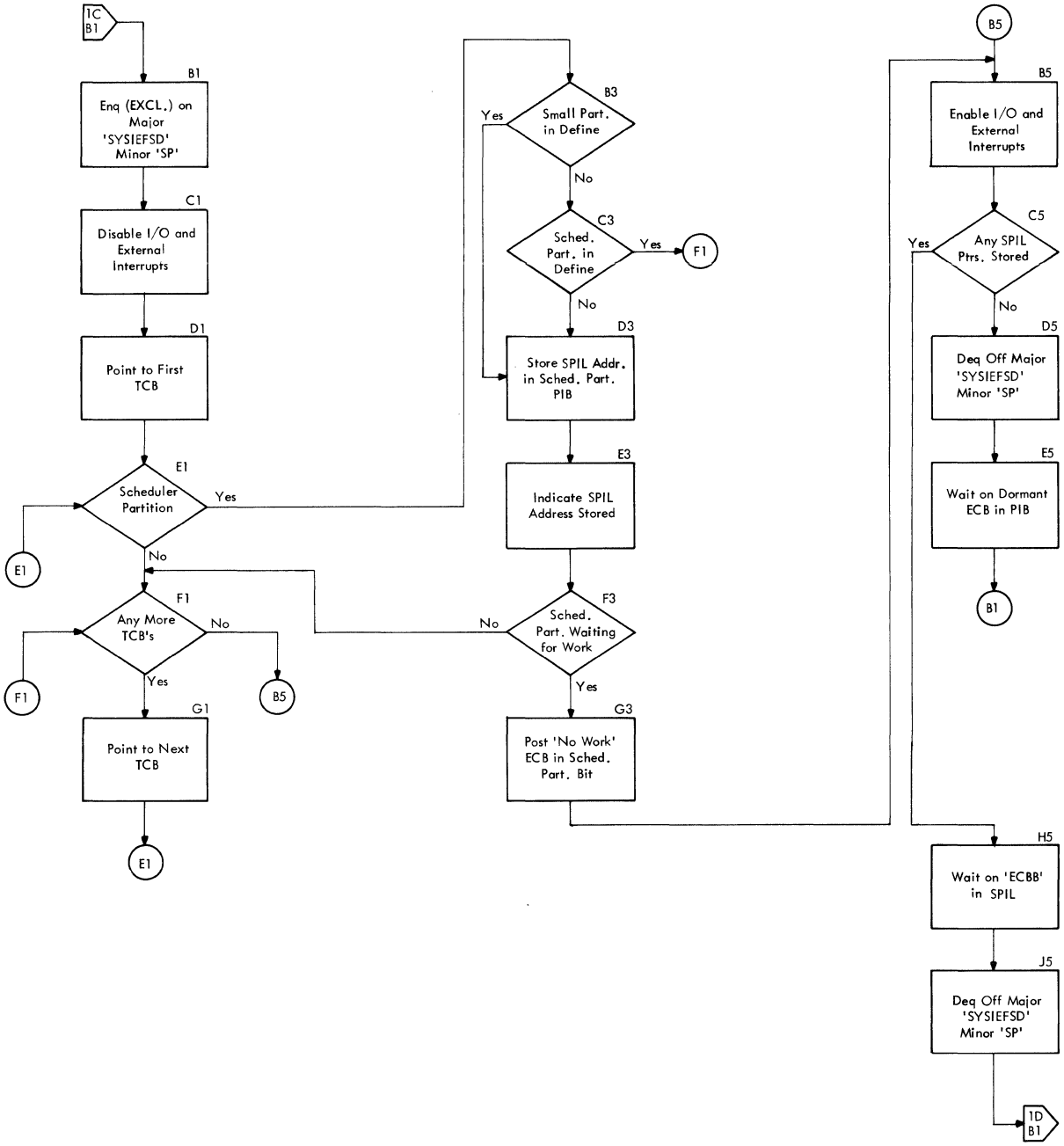


Chart 1D. Small Partition Routine (Part 4 of 4)

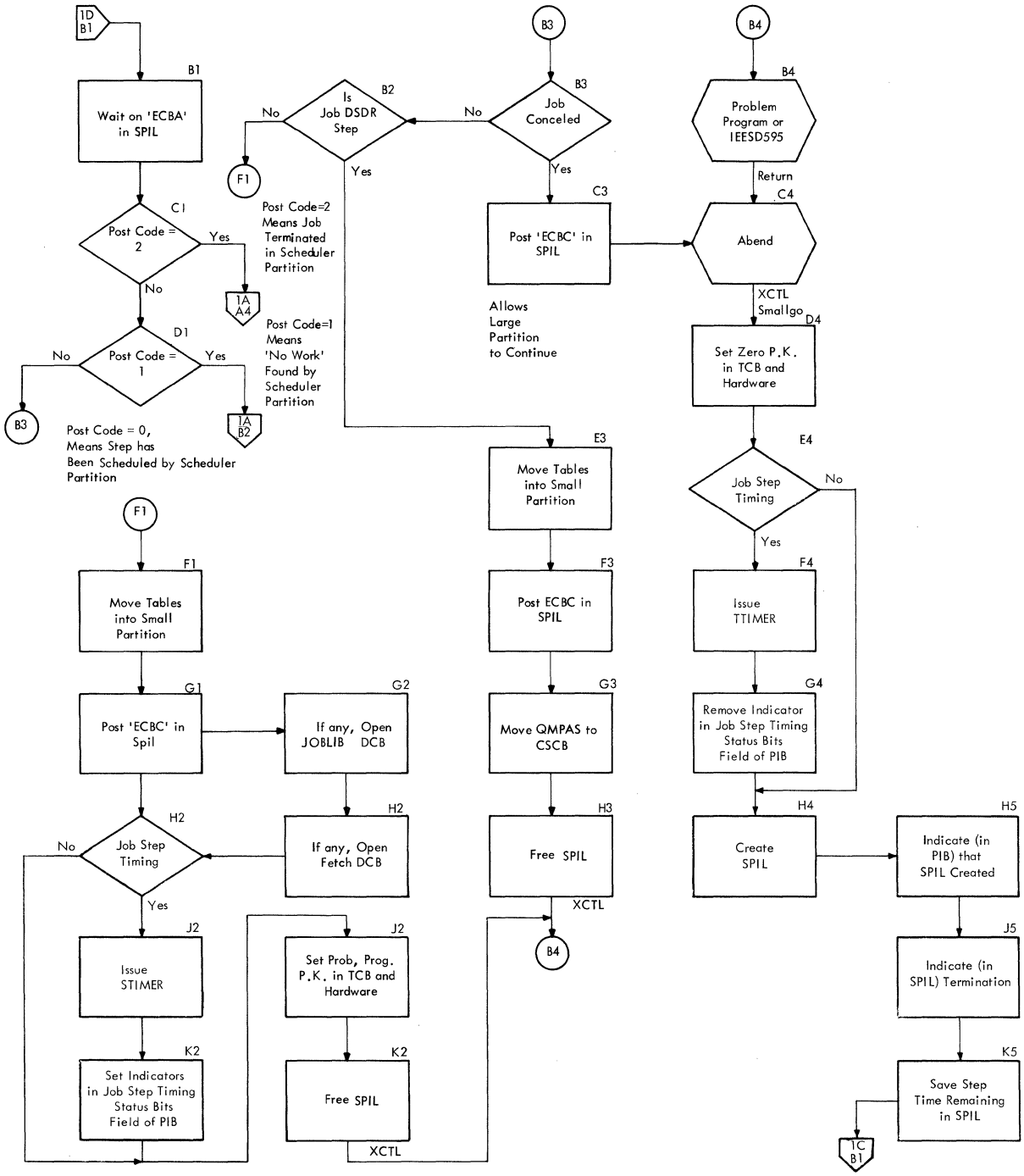


Chart 2. Master Scheduler Task

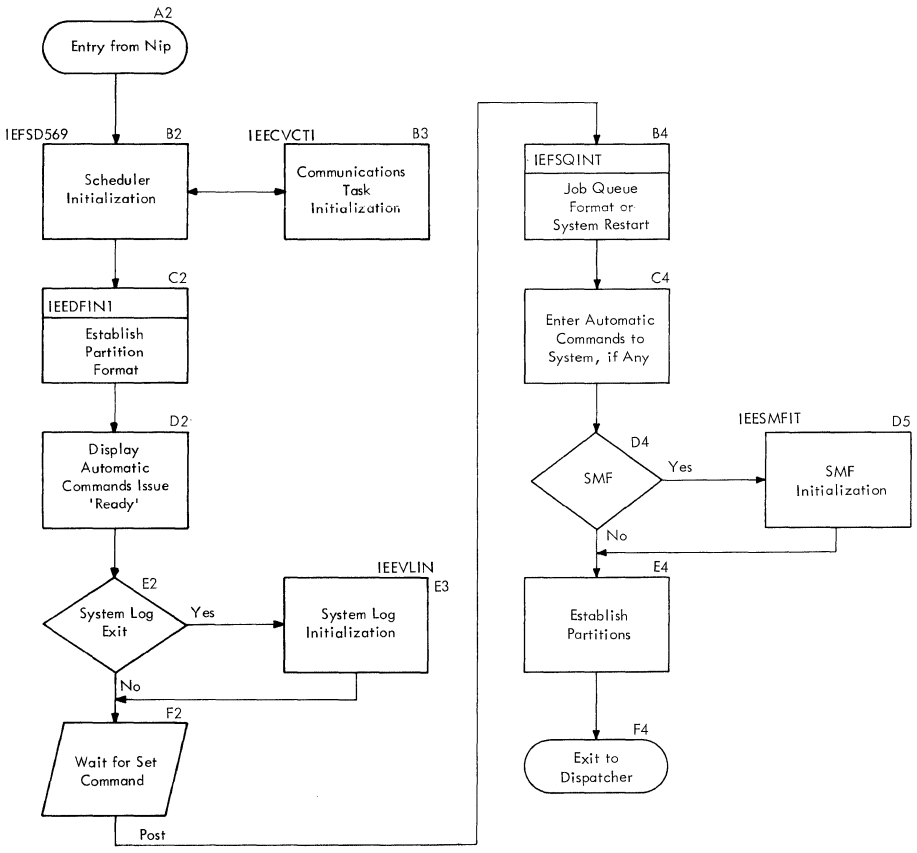


Chart 3A. Queue Alter Express Cancel (Part 1 of 2)

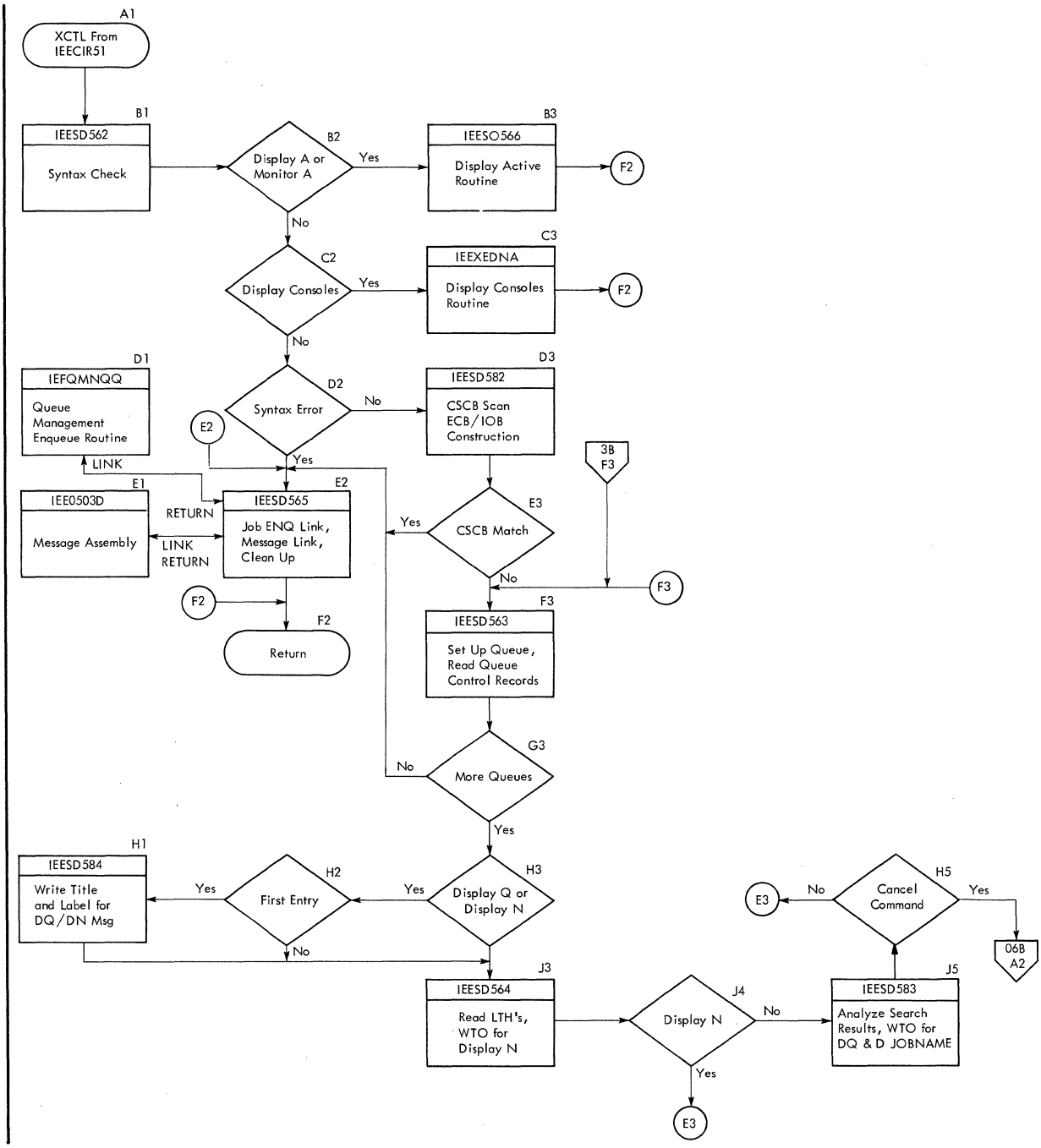


Chart 3B. Queue Alter/Express Cancel (Part 2 of 2)

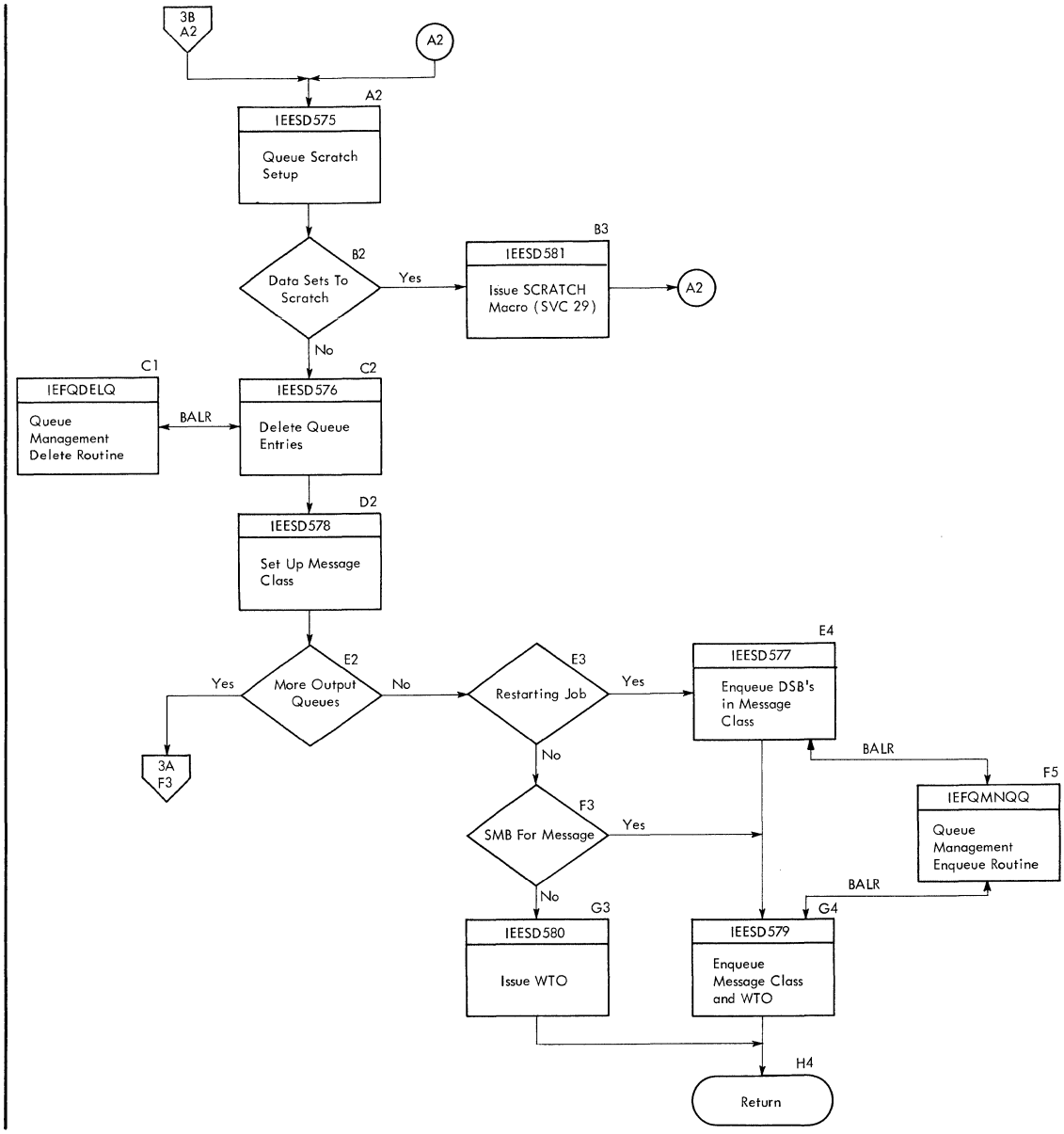


Chart 4. Queue Manager Table Breakup Routine

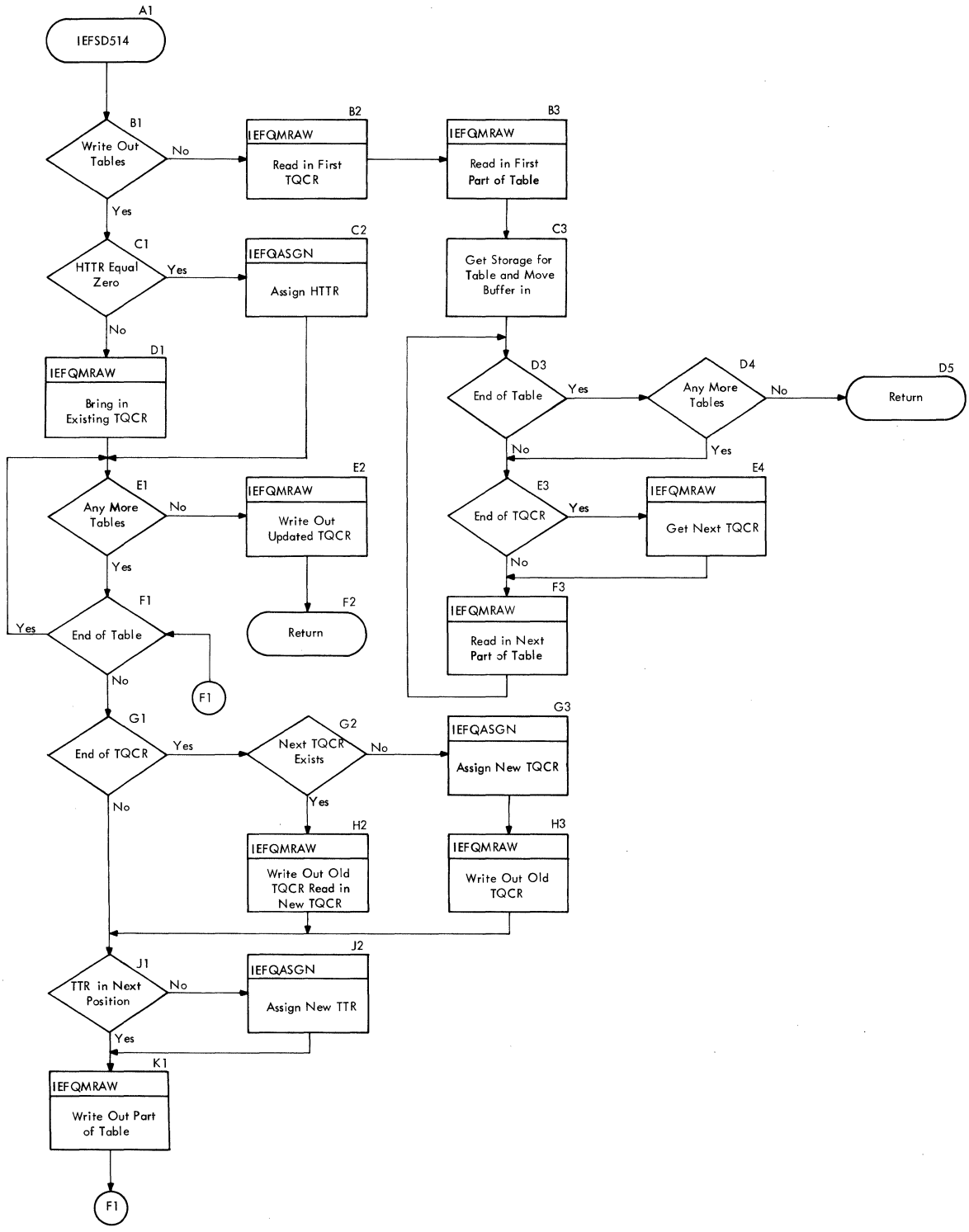


Chart 5. Master Scheduler Resident Command Processor

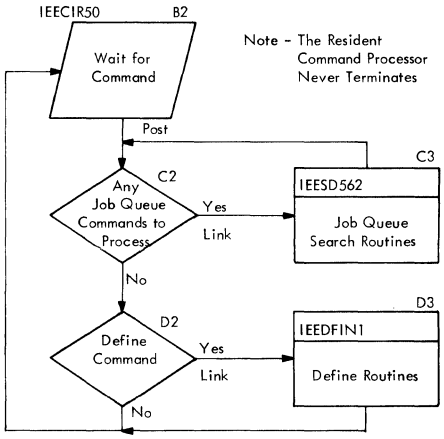
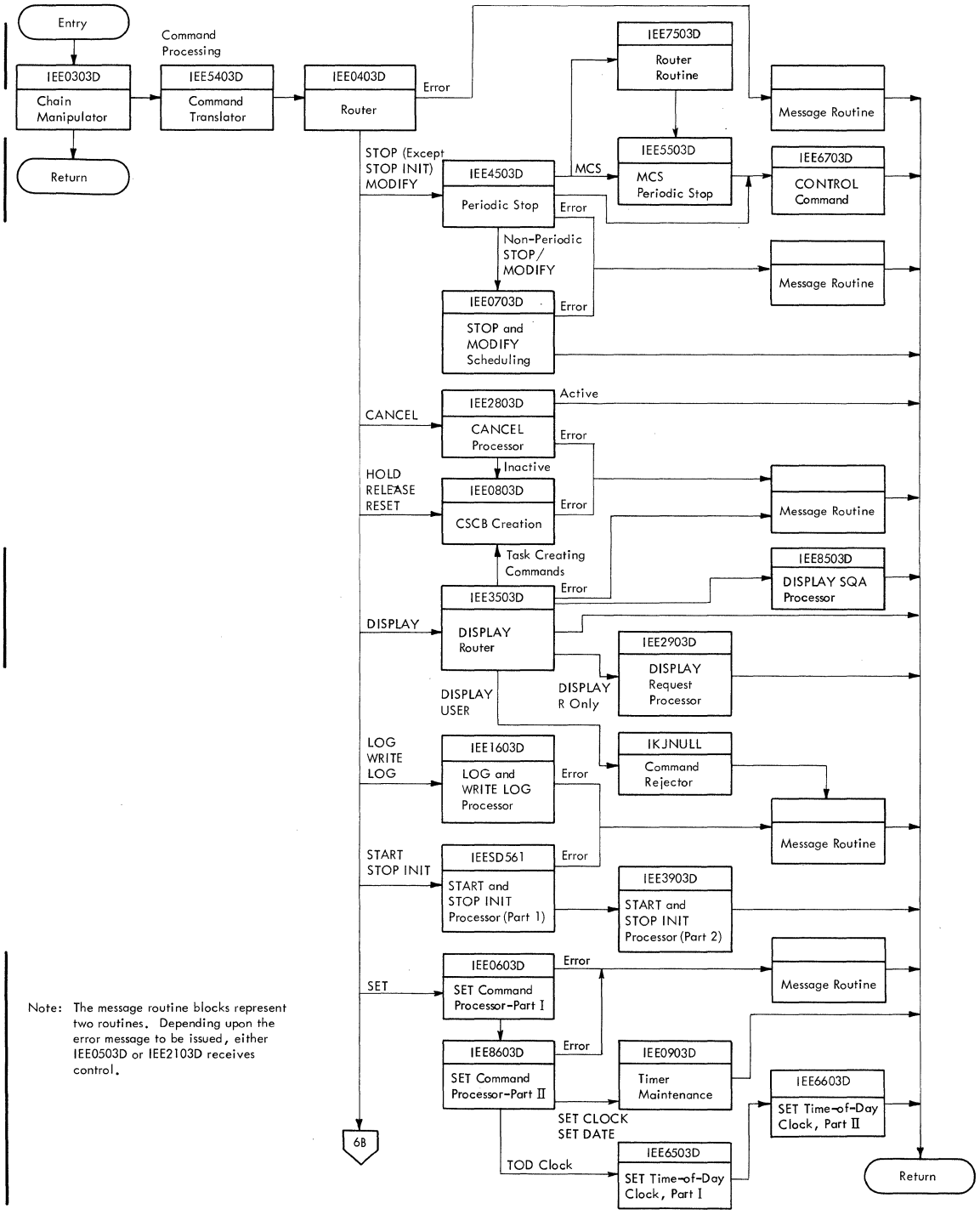


Chart 6A. SVC 34 Command Processing (Part 1 of 4)



Note: The message routine blocks represent two routines. Depending upon the error message to be issued, either IEE0503D or IEE2103D receives control.

Chart 6B. SVC 34 Command Processing (Part 2 of 4)

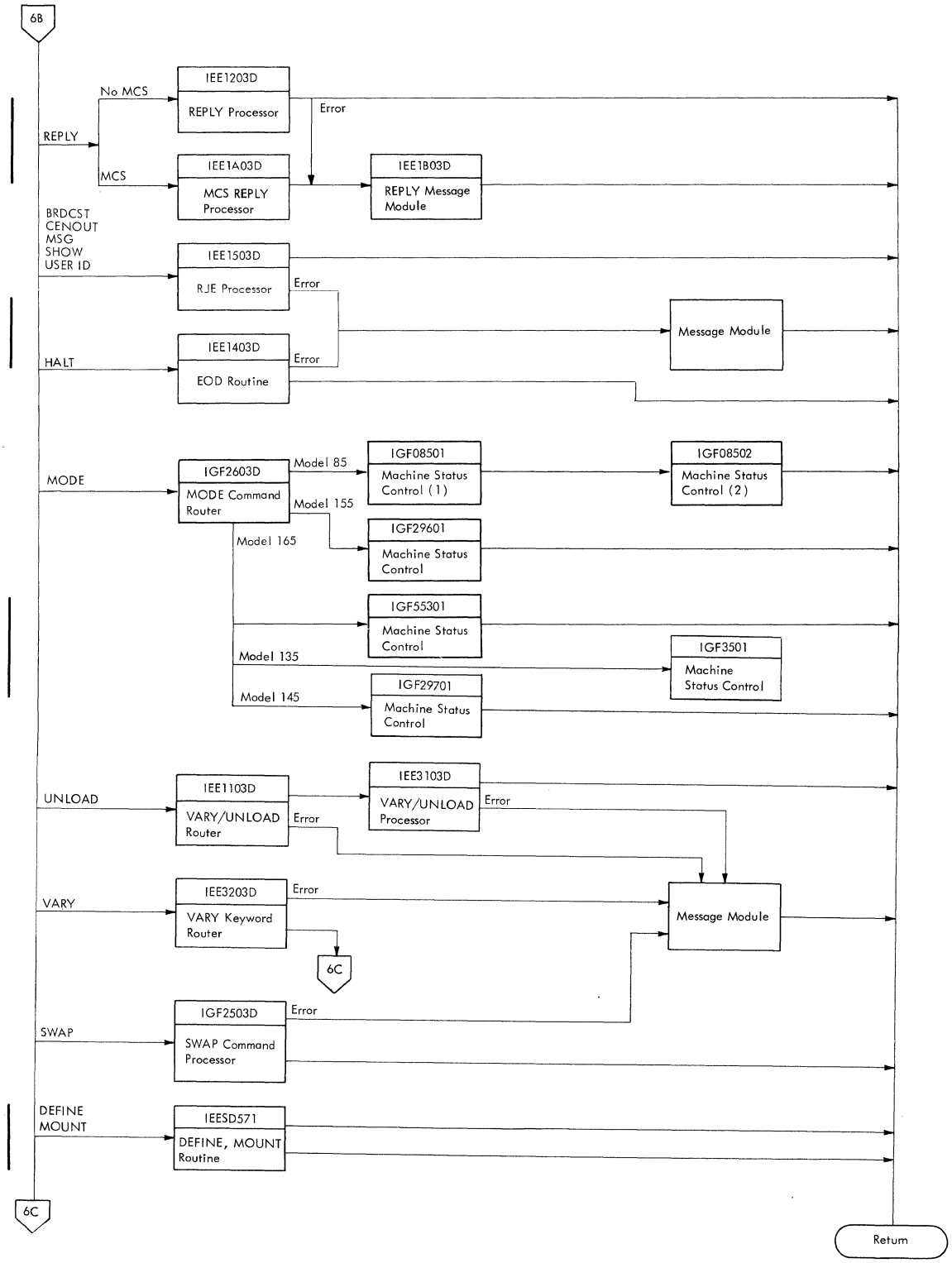


Chart 6C. SVC 34 Command Processing (Part 3 of 4)

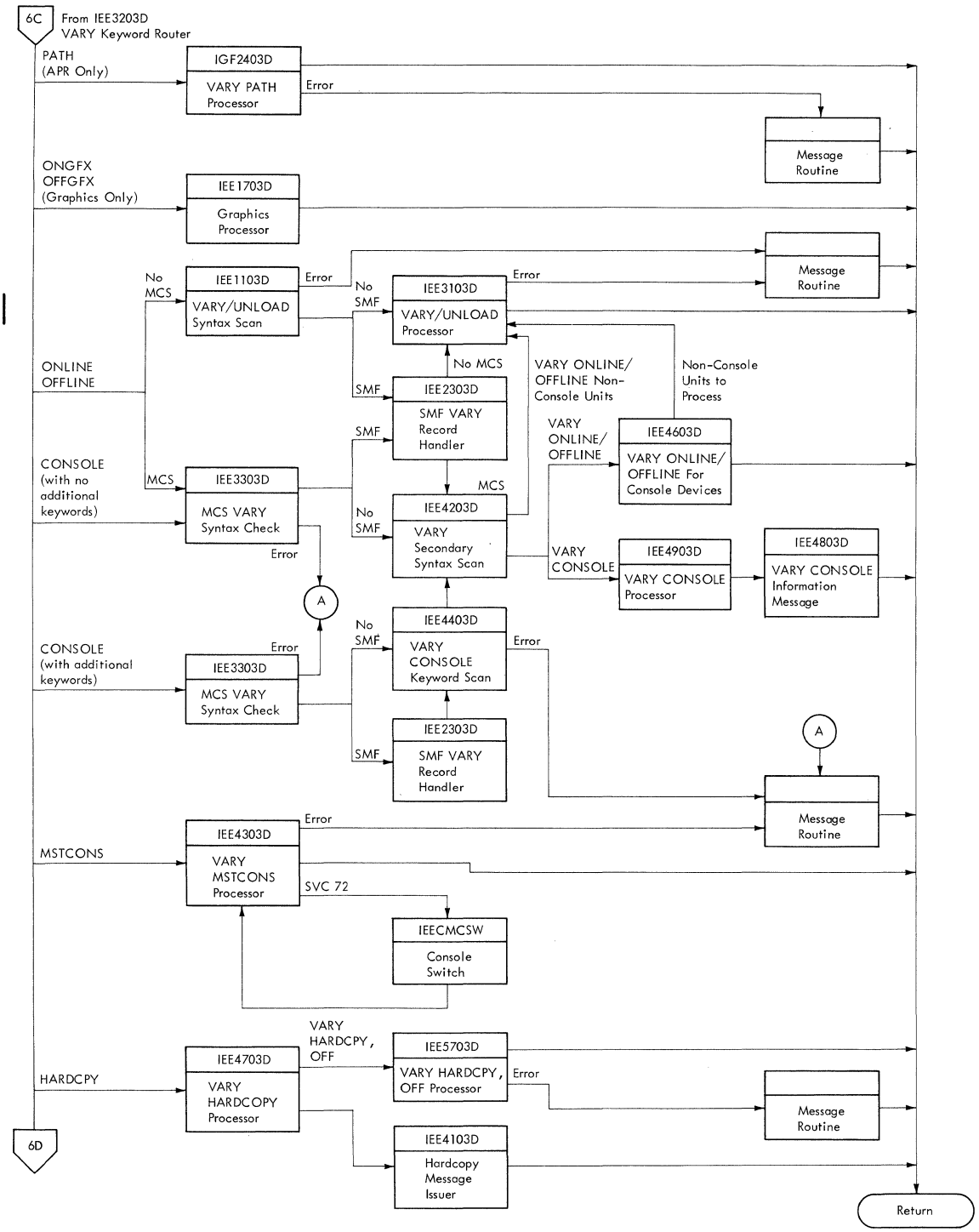


Chart 6D. SVC 34 Command Processing (Part 4 of 4)

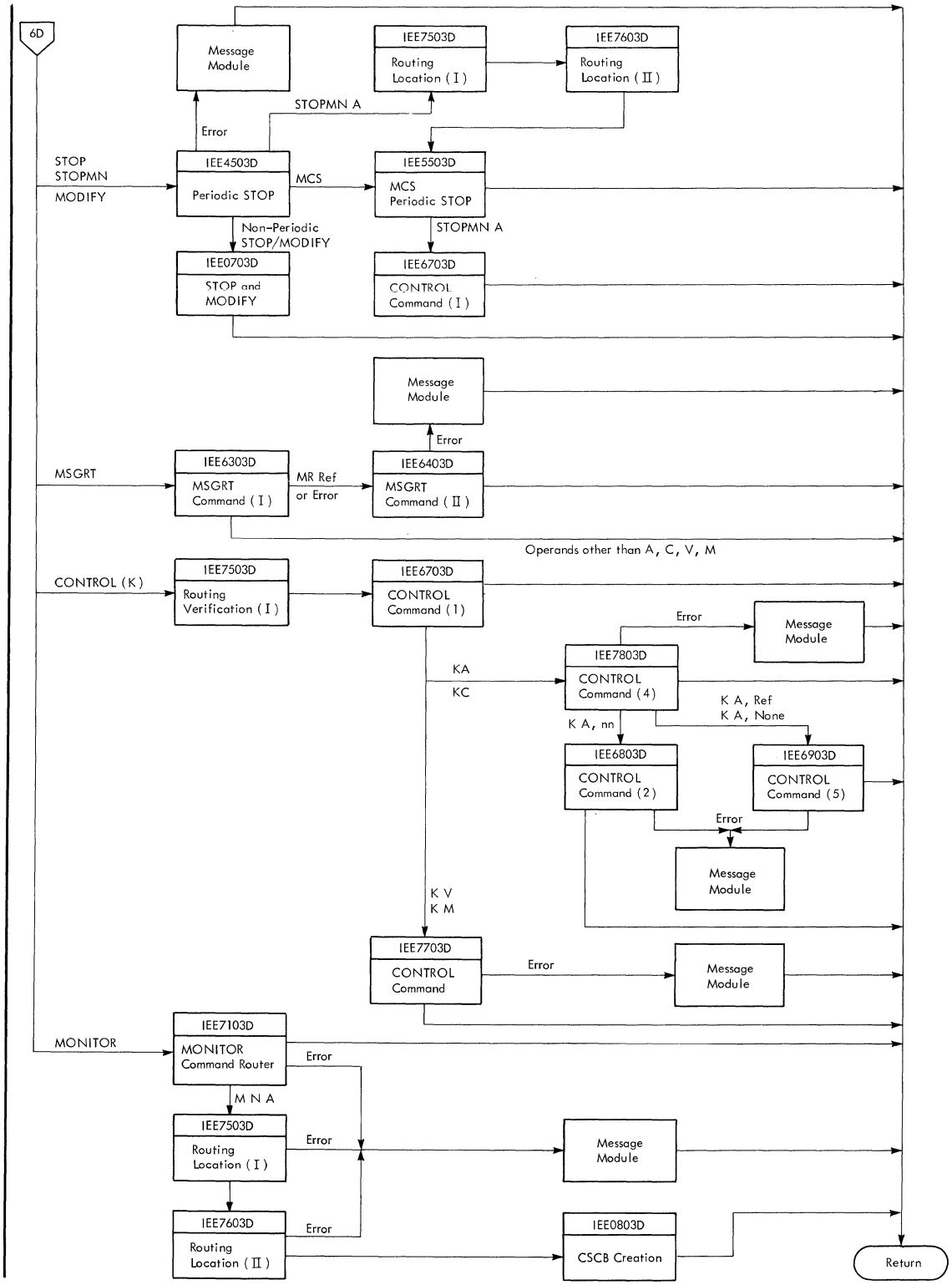


Chart 7. IEFSD518 -- Partition Recovery Routine

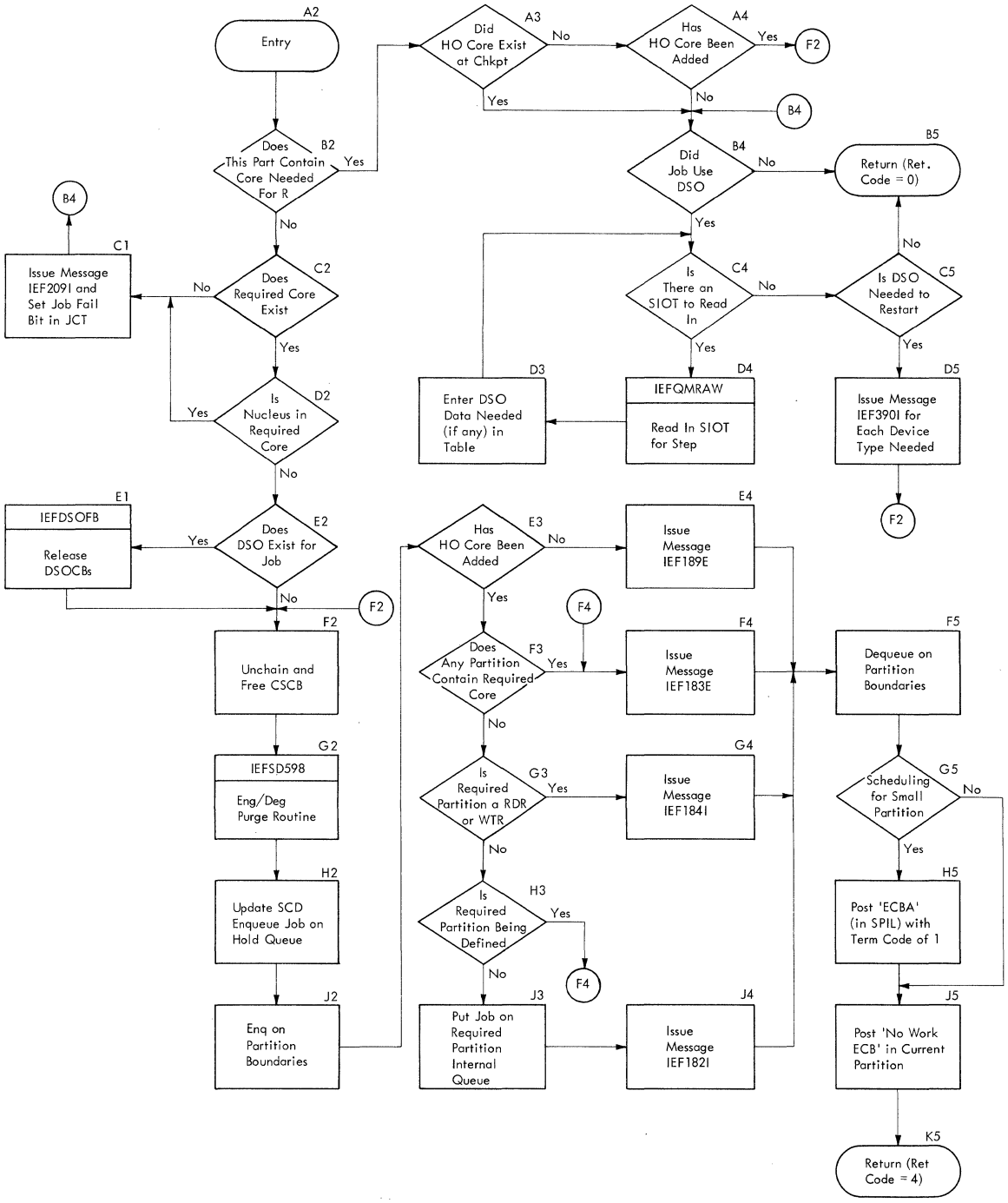


Chart 8. Initiator Control Flow

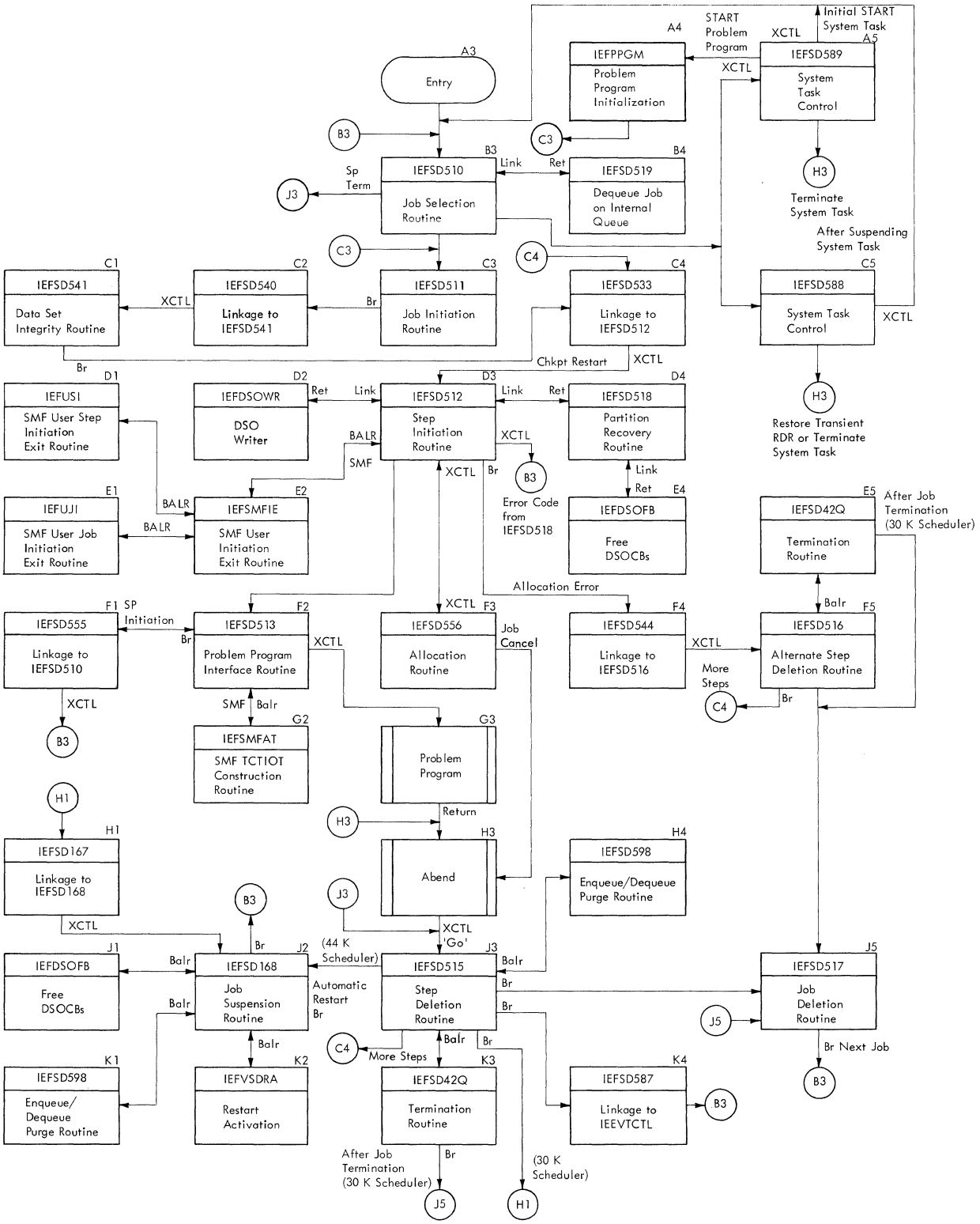


Chart 9A. Job Selection Routine (Part 1 of 5)

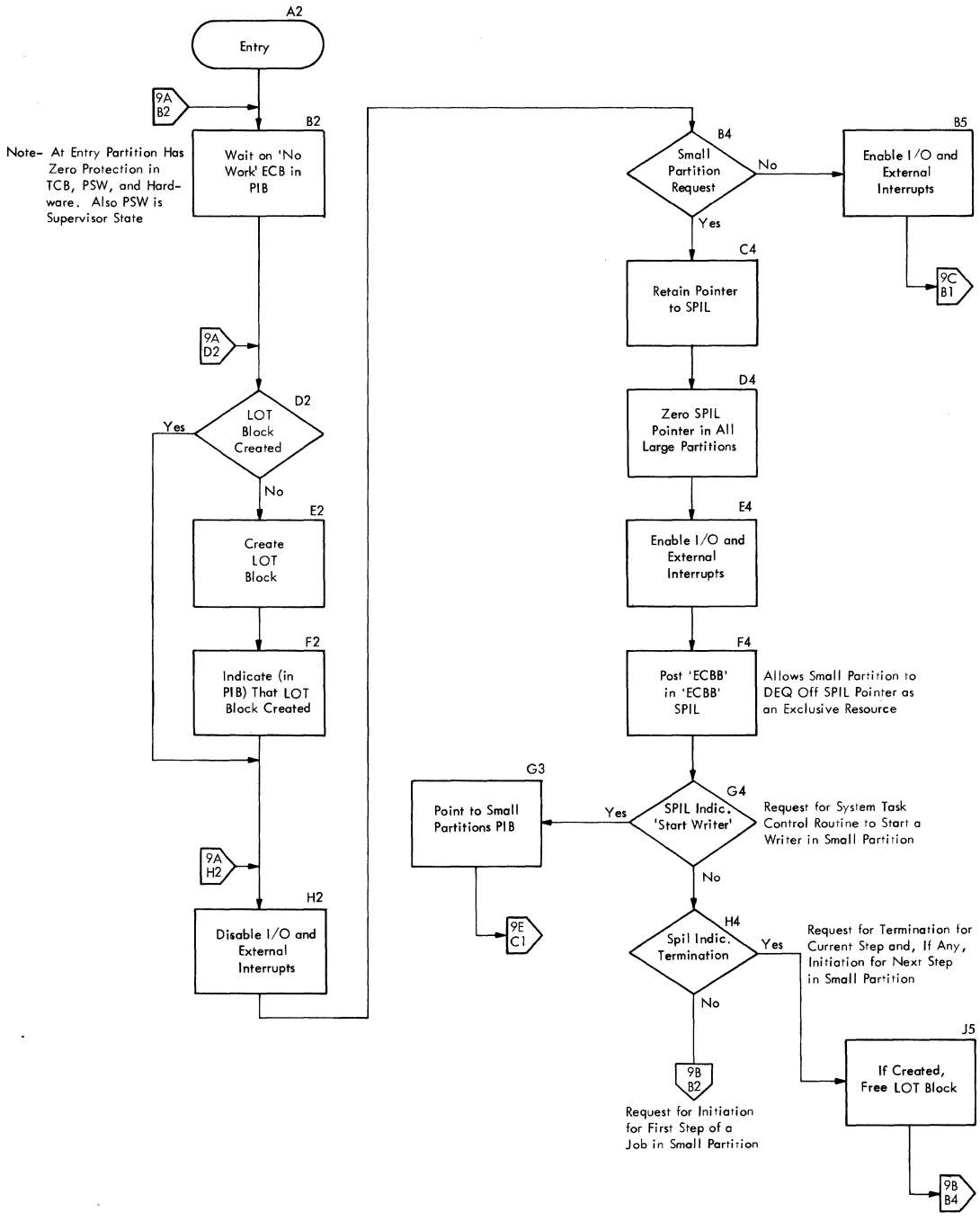


Chart 9B. Job Selection Routine (Part 2 of 5)

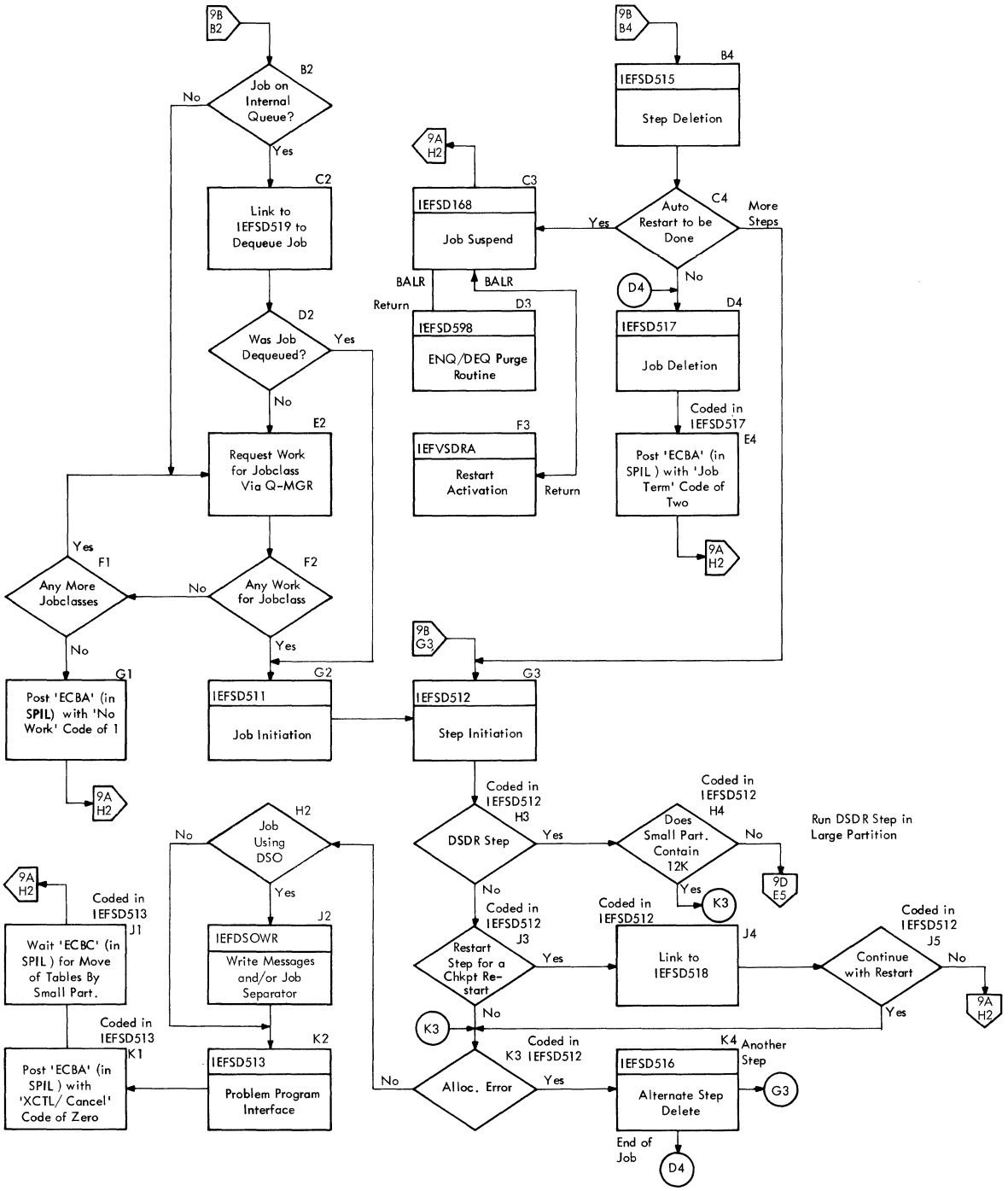


Chart 9C. Job Selection Routine (Part 3 of 5)

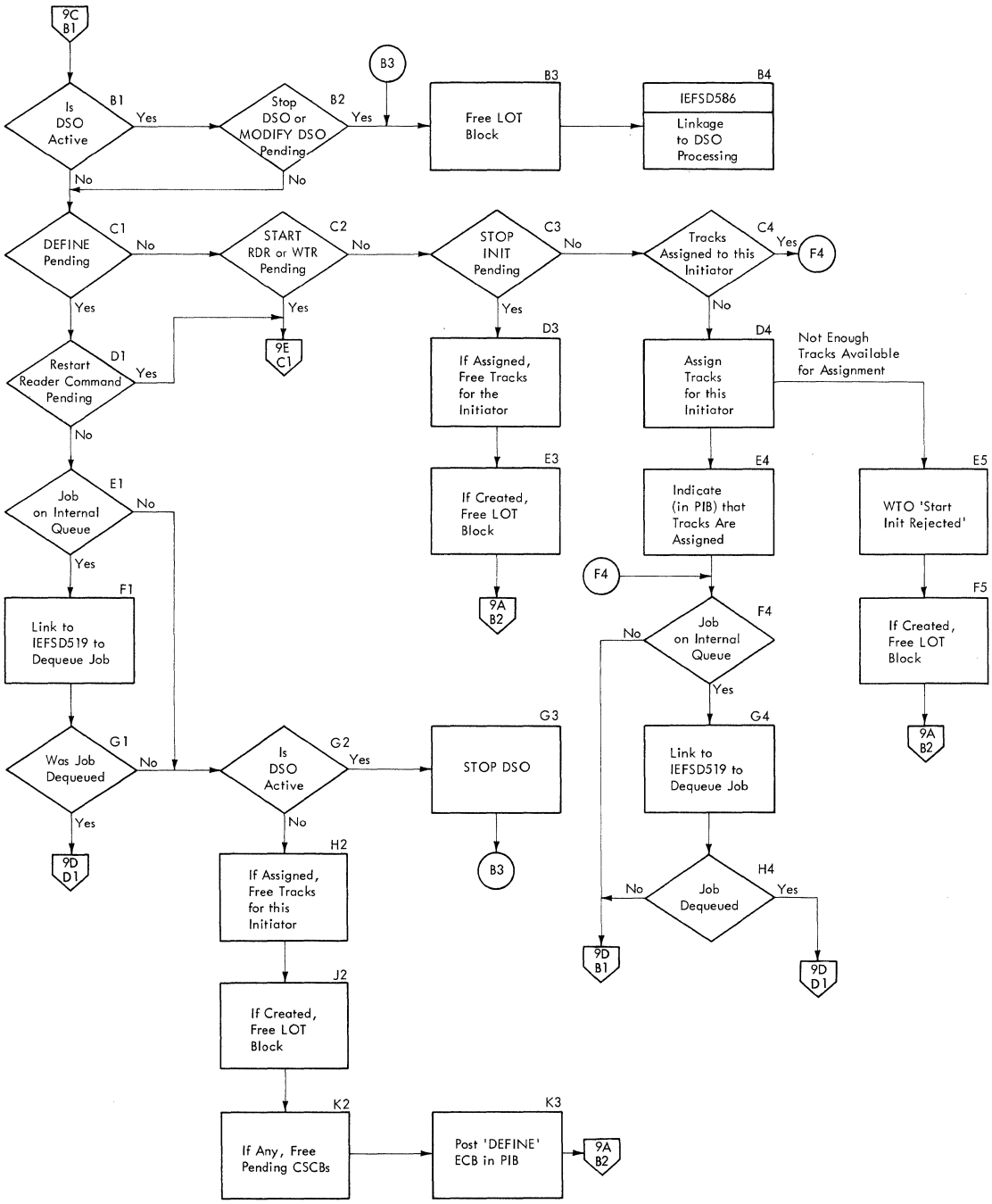


Chart 9D. Job Selection Routine (Part 4 of 5)

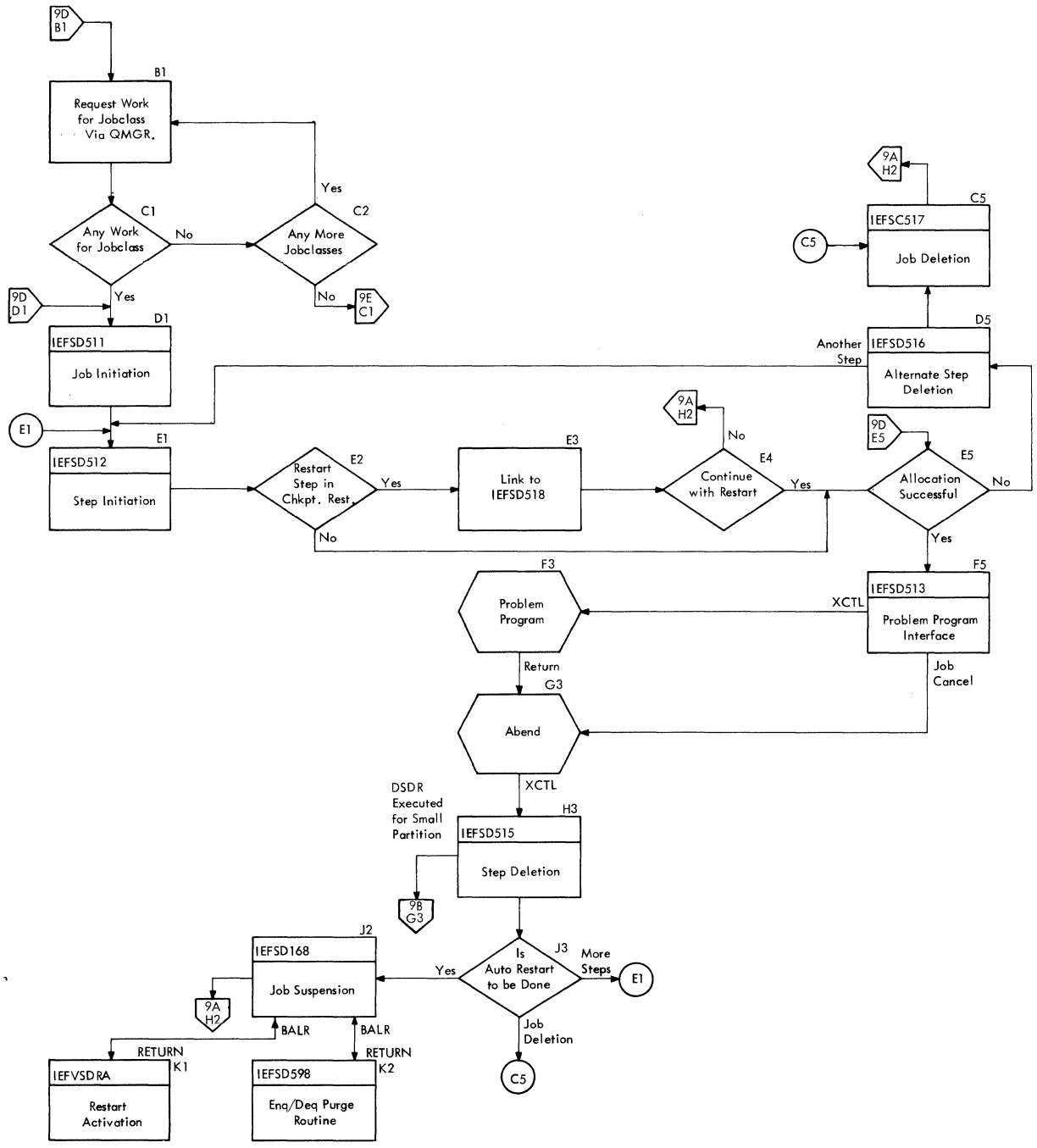
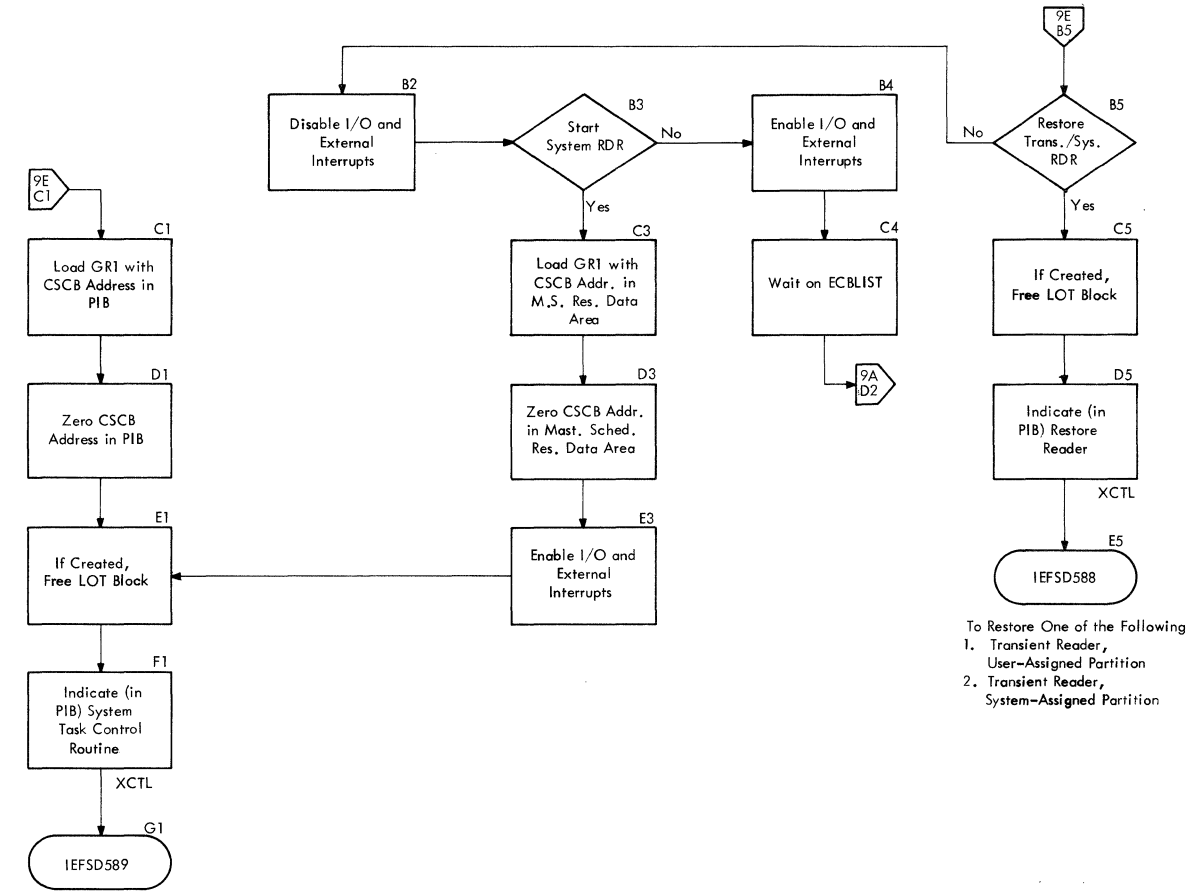


Chart 9E. Job Selection Routine (Part 5 of 5)



Allows System Task Control Routine to Initially Start One of the Following

1. Resident Reader
2. Transient Reader, User-Assigned Partition
3. Transient Reader, System-Assigned Partition
4. Writer, This Partition
5. Writer, Small Partition

To Restore One of the Following

1. Transient Reader, User-Assigned Partition
2. Transient Reader, System-Assigned Partition

Chart 10A. Reader/Interpreter (Part 1 of 3)

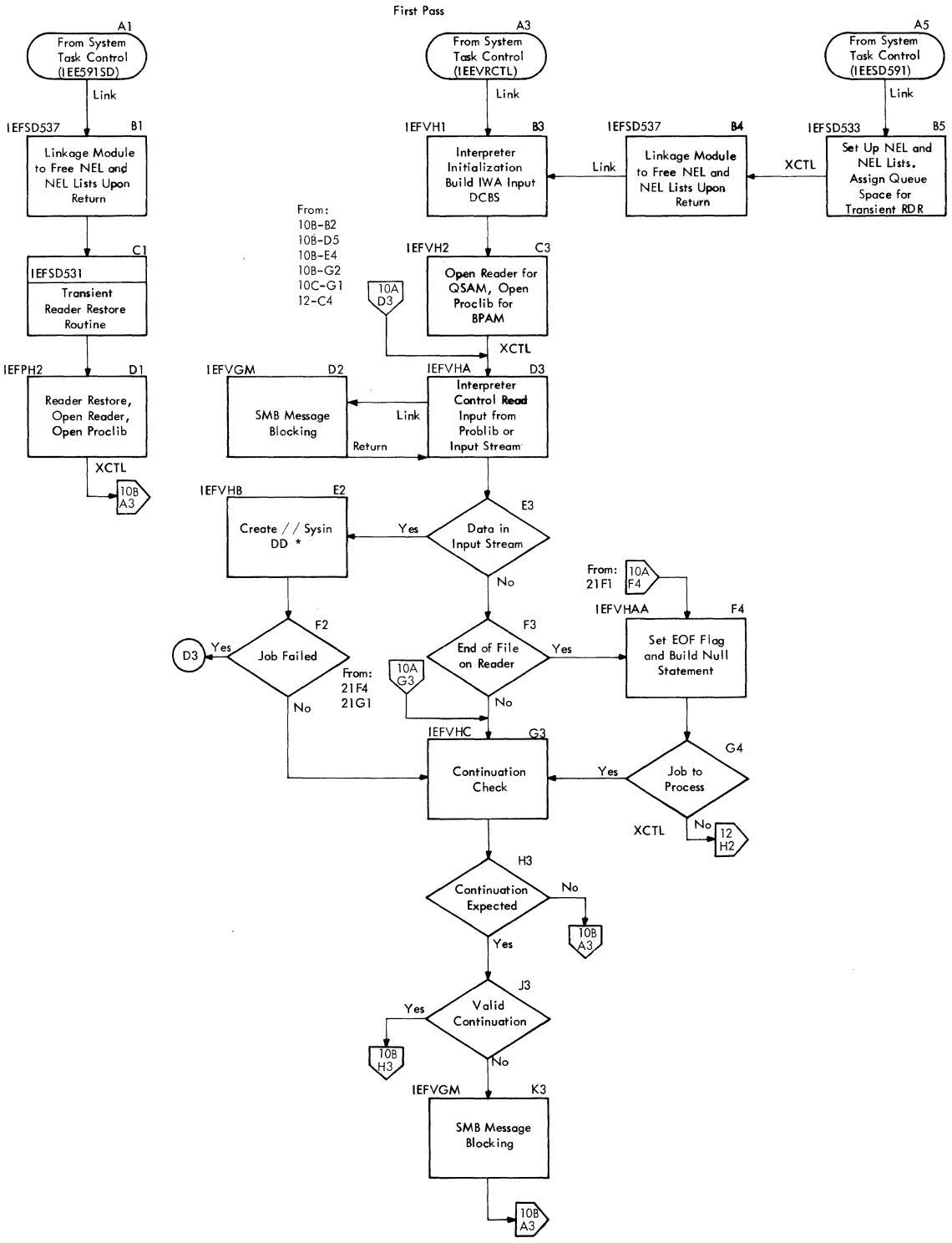


Chart 10B. Reader/Interpreter (Part 2 of 3)

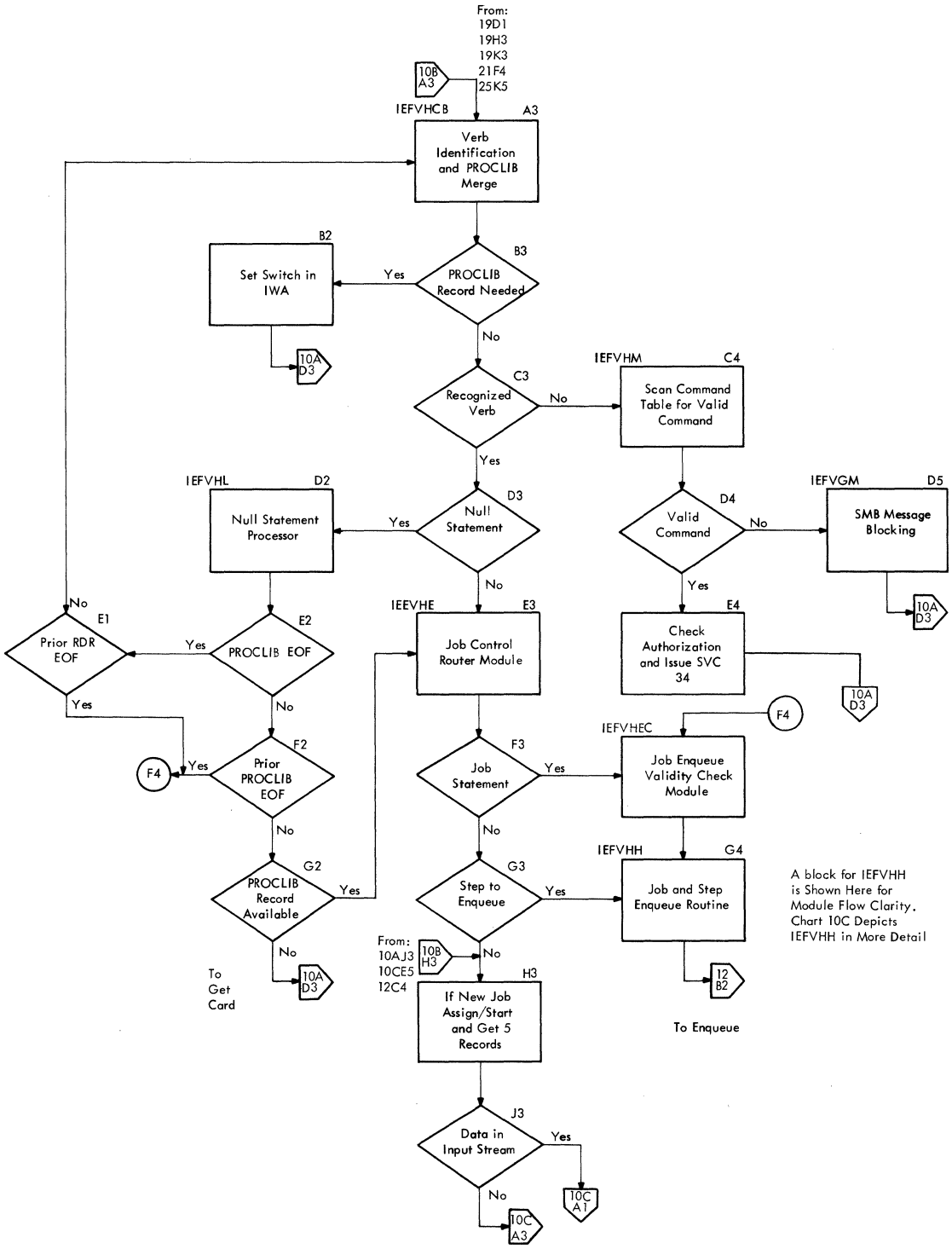


Chart 10C. Reader Interpreter (Part 3 of 3)

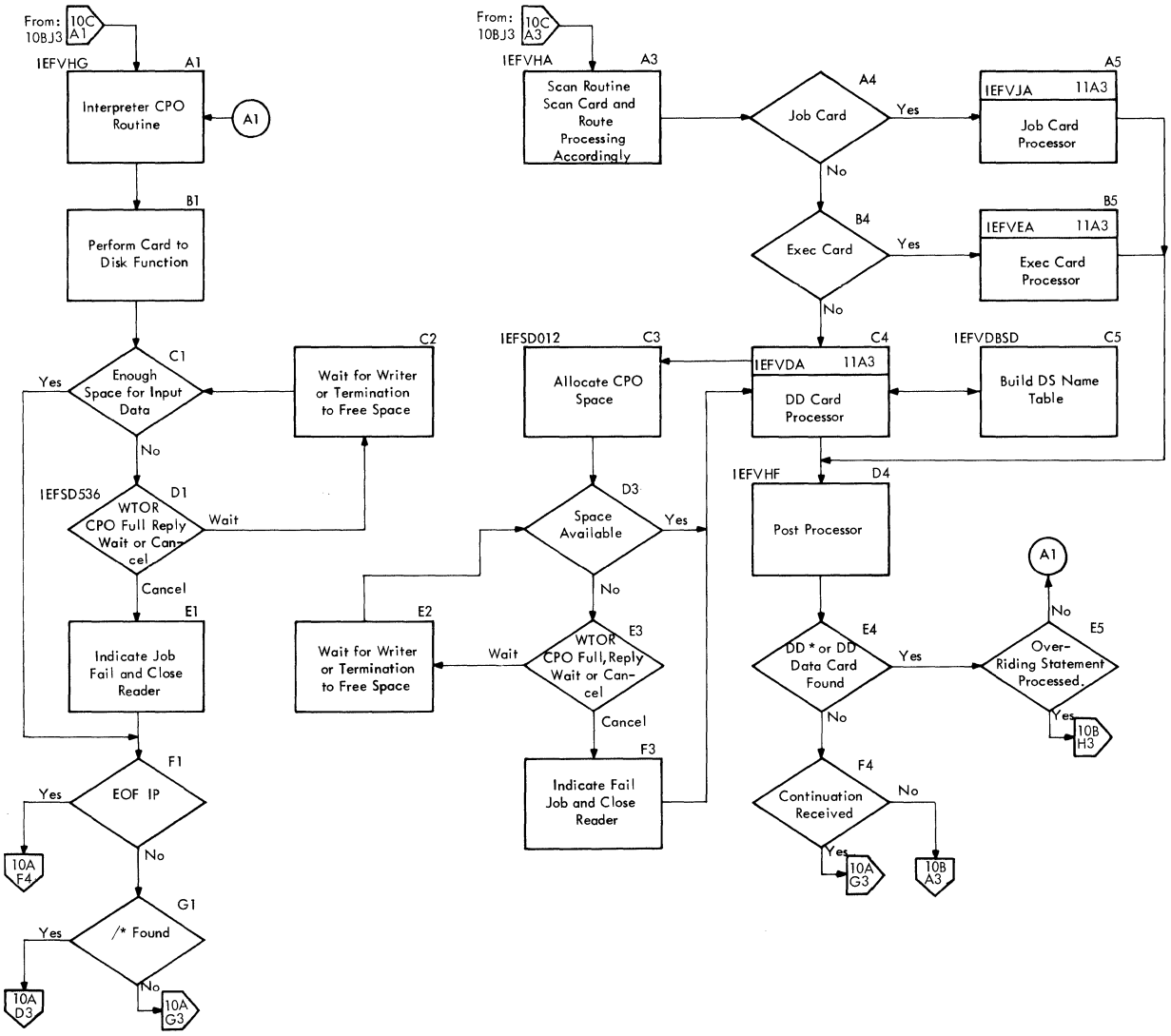


Chart 11. JCL Statement Processor

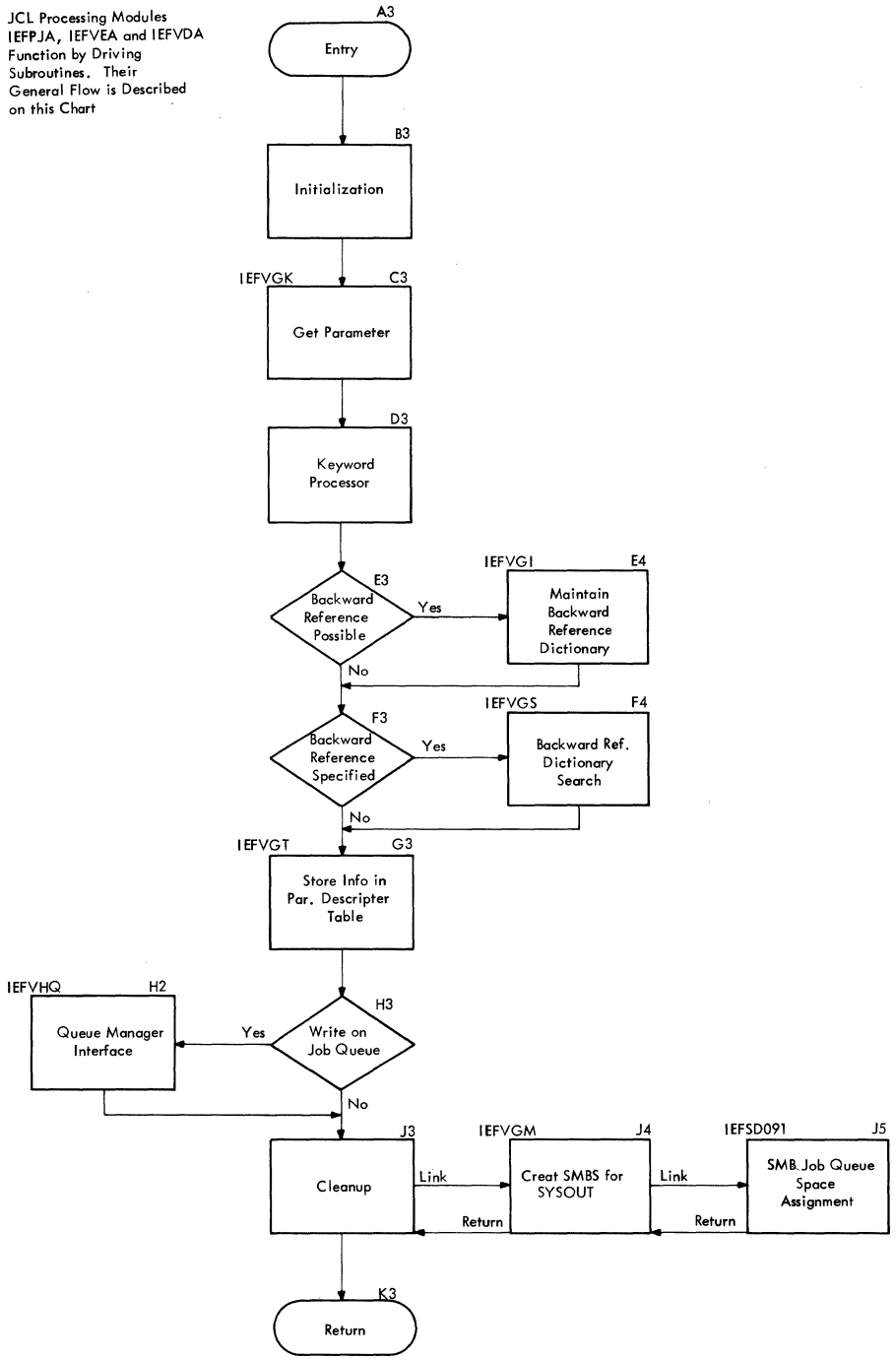


Chart 12. Job and Step Enqueue Routine

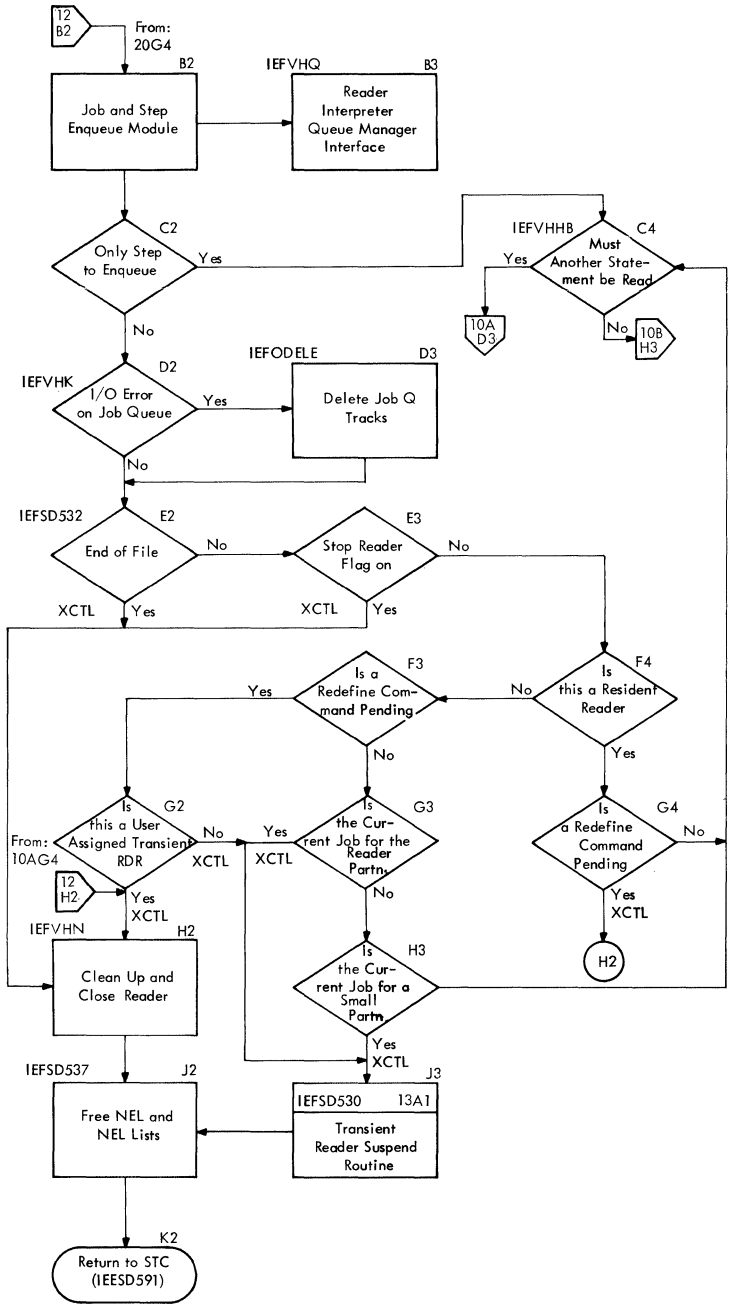


Chart 13. Transient Reader Suspend Routine

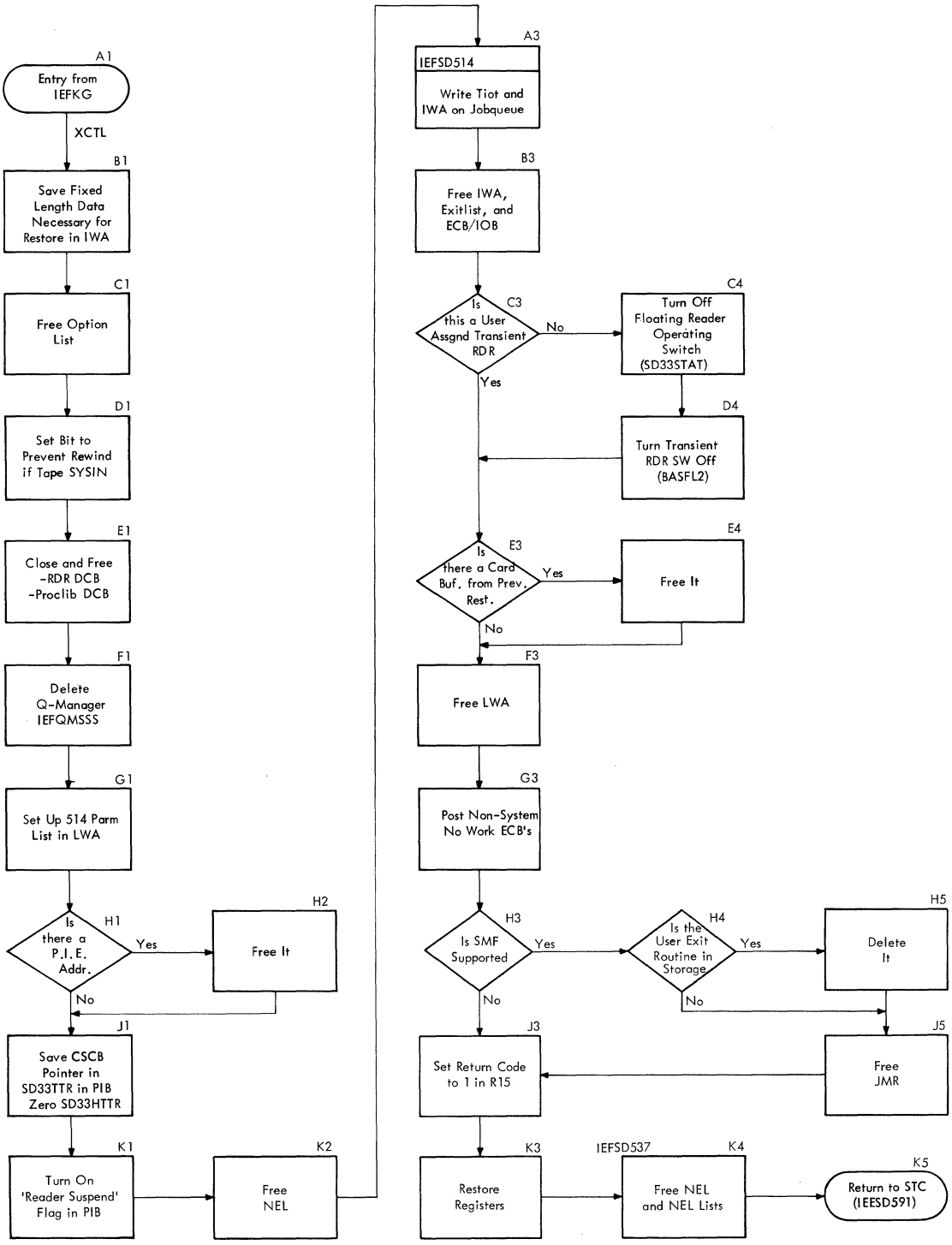


Chart 14. Transient Reader Restore Routine

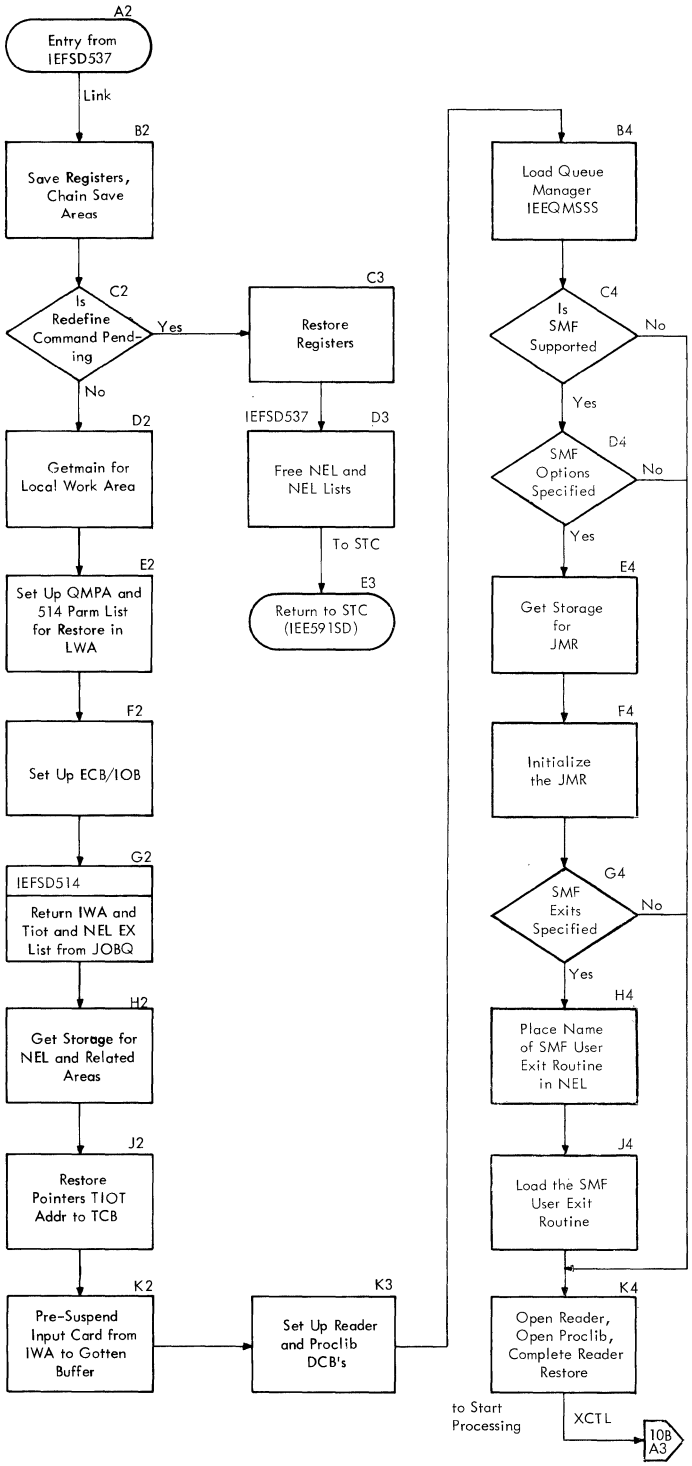


Chart 15. System Output Writer Control Flow

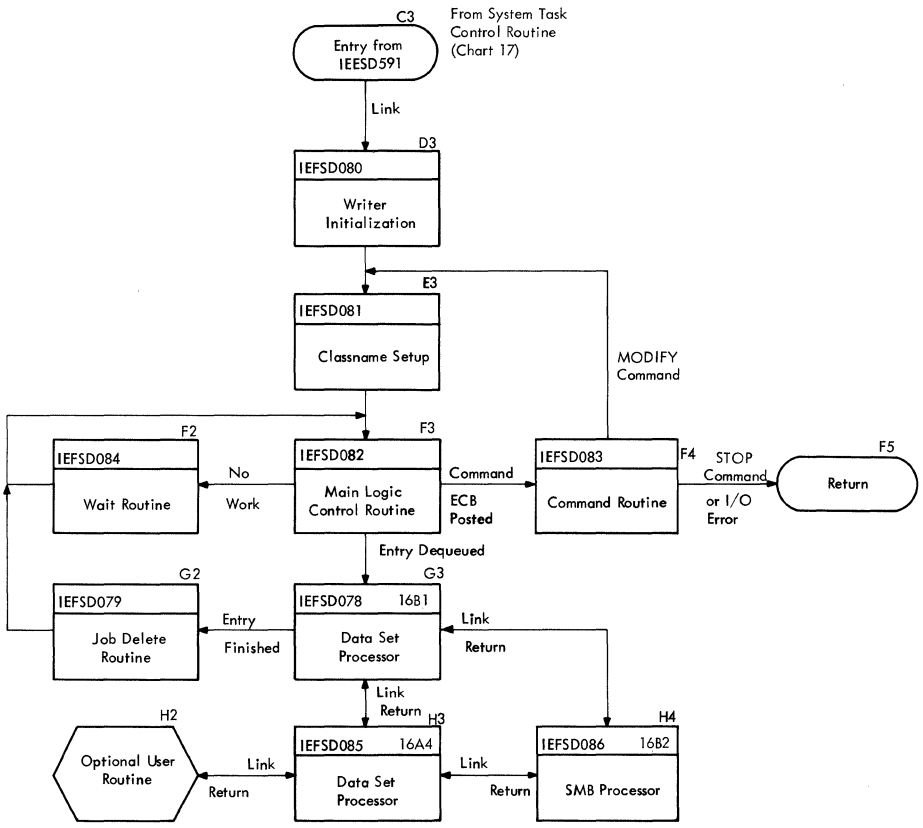


Chart 16. System Output Writer

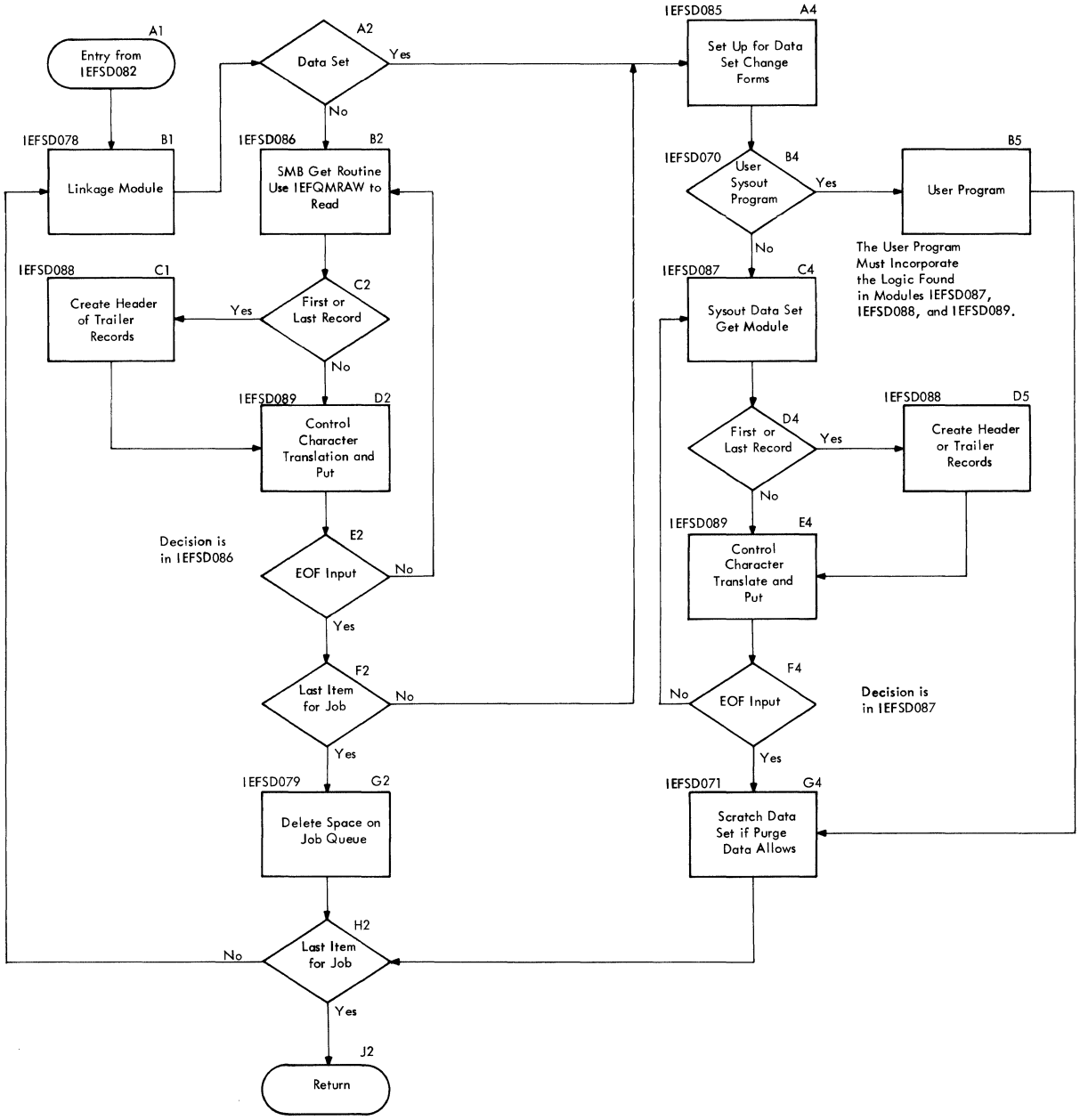
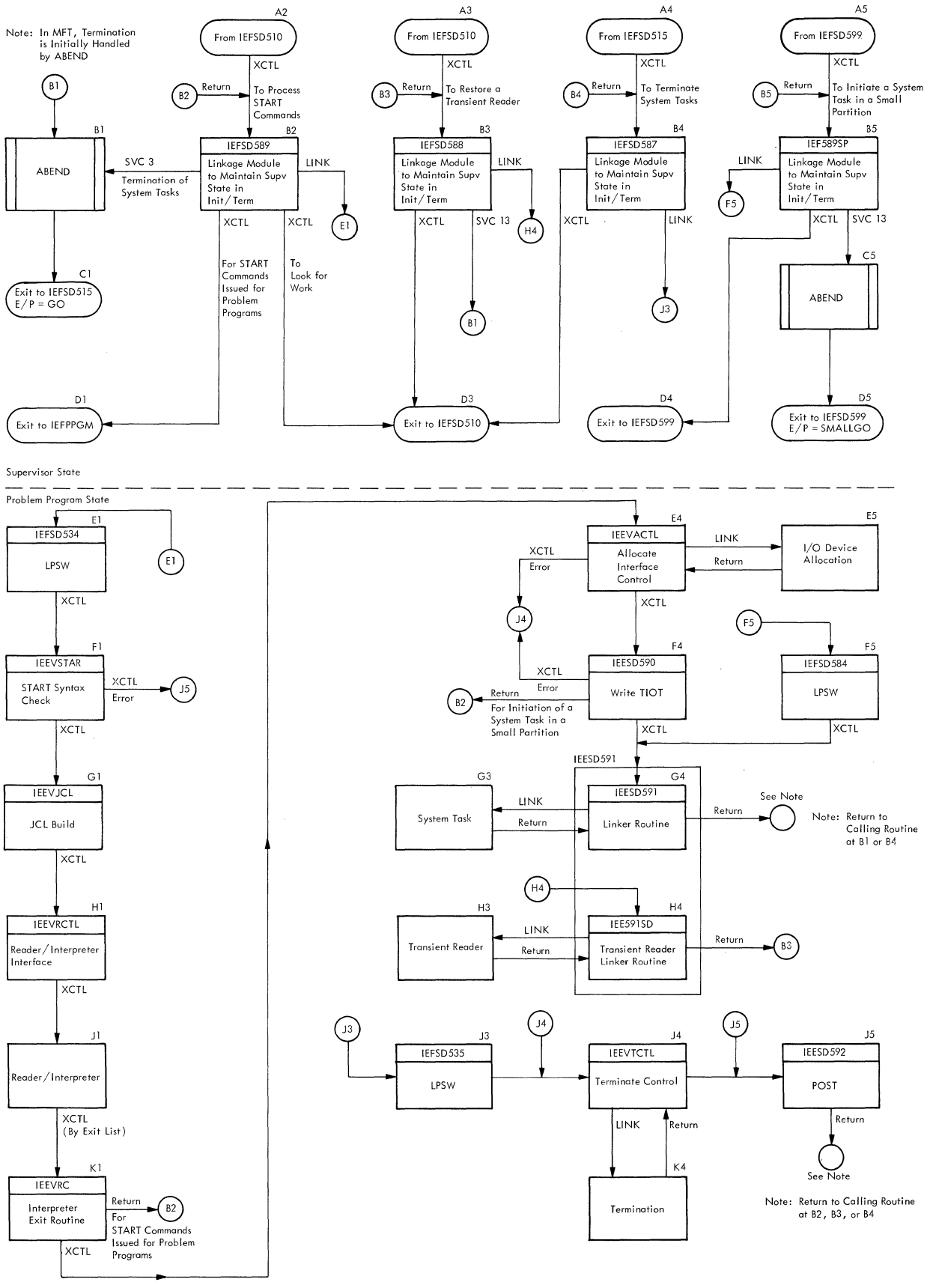


Chart 17. System Task Control



APPENDIX D: DICTIONARY OF ABBREVIATIONS

The following acronyms are used in the text of this manual:

Acronym	Meaning
ACT	account control table
APL	allocate parameter list
APR	alternate path retry
ARSA	allocate register save area
ASB	automatic SYSIN batching
BBX	boundary box
BRDR	background reader
CSCB	command scheduling control block
CVT	communications vector table
DCB	data control block
DCM	display control module
DDR	dynamic device reconfiguration
DEB	data extent block
DSB	data set block
DSDR	data set descriptor record
DSENQ	data set enqueue table
DSO	direct system output
DSOCB	direct system output control block
ECB	event control block
GDG	generation data group
IOB	input/output block
IPL	initial program loading
IWA	interpreter work area
JCL	job control language
JCLS	job control language set
JCT	job control table
JFCB	job file control block
JFCBX	job file control block extension
JMR	job management record
JSEL	job scheduling entrance list
JSOL	job scheduling option list
JSXL	job scheduling exit list
LCT	linkage control table
LOT	life-of-task block
LPL	link parameter list
LTH	logical track header
LSW	local word area
M/S	master scheduler
MCS	multiple console support
MFT	multiprogramming with a fixed number of tasks
MRCT	message routing control table
NEL	interpreter entrance list
NIP	nucleus initialization program
OPT	option buffer
PDQ	passed data set queue
PIB	partition information block
PSW	program status word
QCR	queue control record
QMPA	queue manager parameter area
RB	request block
RJE	remote job entry
SACB	screen area control block
SCD	system output class directory
SCT	step control table
SDT	START descriptor table
SIOT	step input/output table
SMB	system message block
SMCA	system management control area
SMF	system management facility
SPIL	small partition information list
SQA	system queue area
STC	system task control
SVC	supervisor call
SYSIN	system input
SYSOUT	system output
TBR	table breakup routine
TCB	task control block
TCT	timing control table
TCTIOT	timing control task input/output table
TIOT	task input/output table
TQCR	table queue control record
TQE	timer queue element
UPL	user's parameter list
UCM	unit control module
UCME	unit control module entry
WTP	write-to-programmer
WTPCB	write-to-programmer control block
XSA	extended save area

INDEX

Indexes to program logic manuals are consolidated in the publication IBM System/360 Operating System: Program Logic Manual Master Index, GY28-6717. For additional information about any subject listed below, refer to other publications listed for the same subject in the Master Index.

READER'S COMMENT FORM

IBM System/360 Operating System
Job Management with MFT
Program Logic Manual

Order No. GY27-7128-7

Please use this form to express your opinion of this publication. We are interested in your comments about its technical accuracy, organization, and completeness. All suggestions and comments become the property of IBM.

Please do not use this form to request technical information or additional copies of publications. All such requests should be directed to your IBM representative or to the IBM Branch Office serving your locality.

- Please indicate your occupation: _____
- How did you use this publication?
  - ☐ Frequently for reference in my work.
  - ☐ As an introduction to the subject.
  - ☐ As a textbook in a course.
  - ☐ For specific information on one or two subjects.
- Comments (Please include page numbers and give examples.):

- Thank you for your comments. No postage necessary if mailed in the U.S.A.

YOUR COMMENTS, PLEASE . . .

This manual is part of a library that serves as a reference source for systems analysts, programmers and operators of IBM systems. Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Note: Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

Fold Fold

FIRST CLASS
PERMIT NO. 81
POUGHKEEPSIE, N.Y.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY ...

IBM Corporation
P.O. Box 390
Poughkeepsie, N.Y. 12602

Attention: Programming Systems Publications
Department D58

Fold Fold

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

Cut Along Line

System/360 OS Job Management with MFT (S360-36) Printed in U.S.A. GY27-7128-7

System/360 OS Job Management with MFT (S360-36) Printed in U.S.A. GY27-7128-7

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]