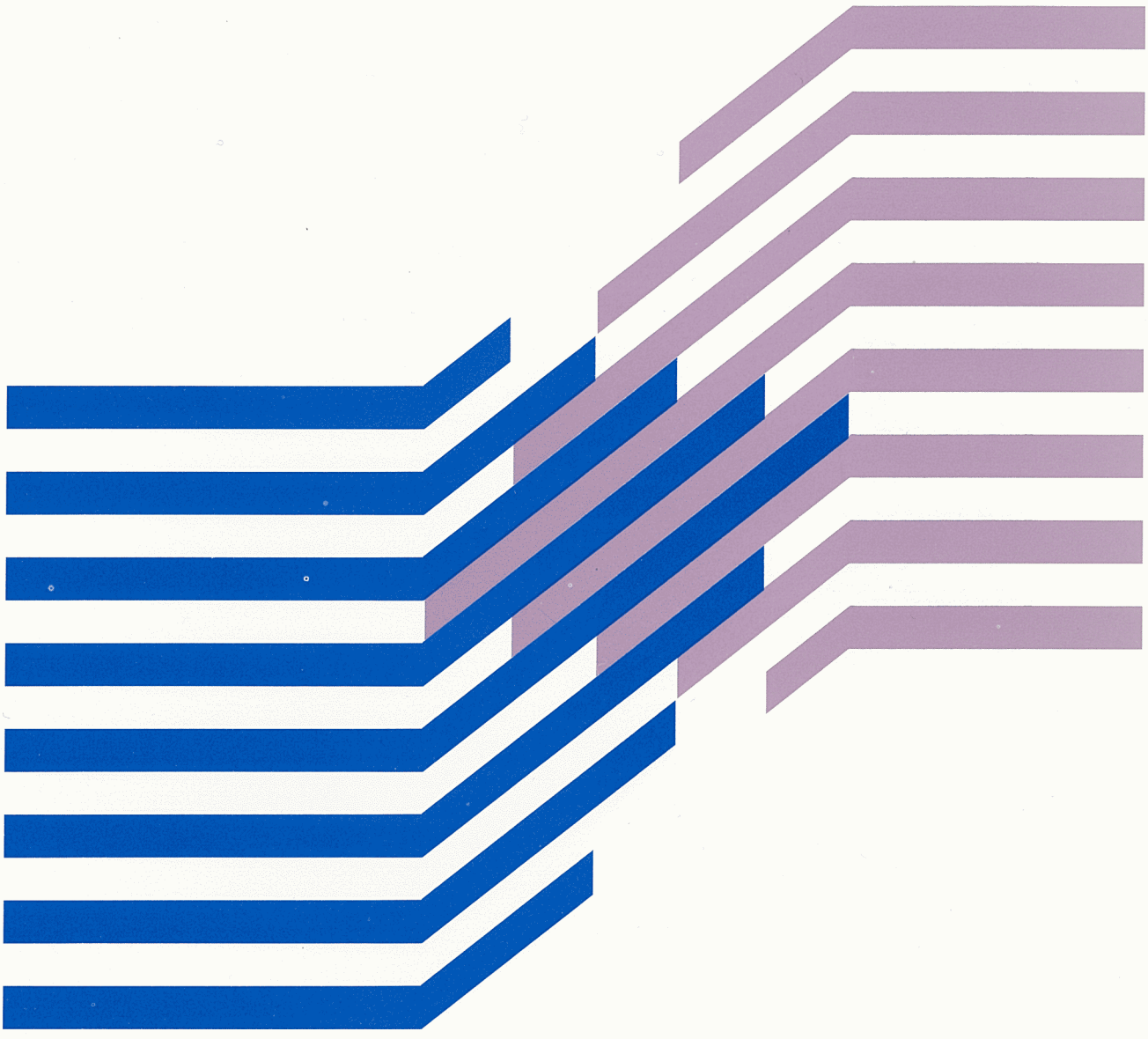
GC28-1828-1

MVS/ESA System Programming Library: Initialization and Tuning

MVS/System Product: JES2 Version 3 JES3 Version 3



GC28-1828-1

MVS/ESA System Programming Library: Initialization and Tuning

MVS/System Product: JES2 Version 3 JES3 Version 3

Second Edition (November, 1988)

This is a major revision of, and obsoletes, GC28-1828-0. See the Summary of Changes regarding new and changed information made to this publication. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

This edition applies to Version 3 of MVS/System Product (5685-001 or 5685-002) and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters. Changes are made periodically to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370 Bibliography*, GC20-0001, for the editions that are applicable and current.

References in this publication to IBM products or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product in this publication is not intended to state or imply that only IBM's product may be used. Any functionally equivalent product may be used instead. This statement does not expressly or implicitly waive any intellectual property right IBM may hold in any product mentioned herein.

Publications are not stocked at the address given below. Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Information Development, Department D58, Building 921-2, PO Box 950, Poughkeepsie, NY 12602. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Summary of Changes

Summary of Changes
for GC28-1828-1
MVS/System Product Version 3 Release 1.0e

This book contains information previously presented in *MVS/Extended Architecture SPL: Initialization and Tuning*, GC28-1828-0. The following summarizes the changes to that information.

New Information:

- Library lookaside facility (LLA) now supports production datasets as well as LNKLST data sets.
- SYS1.PARMLIB member CSVLLAxx is updated with new control statements to support LLA.
  - REMOVE allows you to dynamically remove libraries from the list of libraries managed by LLA.
  - MEMBERS allows you to refresh dynamically specified members in LLA-managed production data sets.
  - -LNKLST- is a shorthand method to specify the whole LNKLST concatenation.
  - PARMLIB(dsn) SUFFIX(xx) allows you to specify another parmlib member that contains more LLA control statements to be processed.
  - FREEZE | NOFREEEZE establishes whether the DASD directory or the LLA maintained directory is used for the LLA-managed data set.
- The GRSCNFxx parmlib member is updated with 3 new parameters.
  - REJOIN indicates whether the system can automatically rejoin the active ring.
  - TOLINT specifies in seconds the tolerance interval for the expected RSA-message to return.
  - ACCELSYS specifies the threshold for GRS ring acceleration.

Changed Information:

- The linklist lookaside facility (LLA) is renamed to library lookaside facility (LLA).
- The SYS1.PARMLIB member CSVLLAxx is updated with a changed control statement to support LLA.

  LIBRARIES list the names of the LNKLST and production libraries to be managed by LLA.
- The value range was increased for the CWSS and PWSS parameters in the IEAIPSxx member.

Deleted Information:

- Using GTF to track sysevents and the input information for each sysevent has been deleted from *Initialization and Tuning* and moved to *Component Diagnosis and Logic: System Resource Manager, LY28-1592.*

Summary of Changes
for GC28-1828-0
MVS/System Product Version 3 Release 1.0

This book contains information previously presented in *MVS/Extended Architecture SPL: Initialization and Tuning*, GC28-1149-5. The following summarizes the changes to that information.

New Information

Information to support MVS™ System Product Version 3 includes

- A new parameter is added to the SVCPARM statement in IEASVCxx. The new parameter supports SVCs issued in access register ASC mode.
- Increase to the value range for the CWSS and PWSS parameters in the IEAIPSxx member.
- A new system library, SYS1.MIGLIB, to be concatenated after SYS1.LINKLIB
- A new SYS1.PARMLIB member COFVLFxx to support virtual lookaside facility (VLF).
- A new SYS1.PARMLIB member CSVLLAxx to support selective refresh of the linklist lookaside (LLA).
- A new SYS1.PARMLIB member EXSPATxx for automatic responses to certain types of excessive spin loops.
- New IPCS support is available in the BLSCECT member of SYS1.PARMLIB.
- Nine new parameters are available to assist in detecting missing interrupt conditions. See the IECIOSxx member.
- A new parameter in IEASYSxx, NSYSLX, allows you to specify the number of slots in the system function table to be reserved for system linkage indexes.
- A new sysevent AVAILPUP (49 hex) to reserve processor storage frames for dumping services.
- A new sysevent CPUTCONV (4A hex) to convert CPU seconds to service units.

  Information to support MVS/ESA™ DFP Version 3 Release 1.0 when it becomes available includes

  - Support for a new record type in the IEFSSNxx member
  - A new SYS1.PARMLIB member, IGDSMSxx, which contains the parameters that initialize the storage management subsystem and define it to the system.
  - Descriptions of the new SET SMS and SETSMS commands that change the definitions of the storage management subsystem.

  The edition also contains MVS/SP™ Version 3 support that will be effective with MVS/ESA DFP Version 3 Release 1.0.. This information includes a new way of using the VATLSTxx member to define your DASD volumes to the system:

  - Use the new VATDEF statement to establish the installation's default use attributes for permanently-resident DASD volumes.
  - Use generic volume serial numbers and device types for groups of DASD volumes that are mounted on devices that have the same device type and the same mount and use attributes.

Changed Information

The following information is changed for MVS/SP Version 3

- The value range for the SRV keyword in the IEAIPSxx member increases to 999999999.
- Clarification and updates are included for the GTFPARM member.

Deleted Information

MVS/SP Version 3 does not support print dump. References to print dump, AMDPRDMP, have been removed.

About This Book

This book is effectively two manuals in one: a system initialization SRL and a preliminary tuning guide for MVS™. These two "manuals" are combined because of the need for heavy cross referencing between performance discussions and the descriptions of parameters and parmlib members that affect performance. The book describes how to initialize the system, and how to get improved system performance.

Although this book is basically a guide, it also includes elements of logic for certain components: the system resources manager (SRM) and the auxiliary storage manager (ASM). The logic elements are provided either to clarify recommendations and "how it works" descriptions, or to provide a basis for possible internal modification. This statement is especially true of the chapter on SRM.

Trademarks

The following are trademarks of International Business Machines Corporation:

- MVS™
- MVS/ESA™
- NetView™

Who This Book Is For

This book is intended for anyone who defines the initialization parameters for an installation and then fine tunes the system. Usually, this person is a systems programmer. The book assumes that the reader can:

- Code JCL statements to execute programs or cataloged procedures.
- Code in assembler language and read assembler, loader, and linkage editor output.

How This Book Is Organized

This book has six parts.

- Part 1: Introduction
- Part 2: Storage Management Overview
- Part 3: System Initialization
- Part 4: Auxiliary Storage Management Initialization
- Part 5: The System Resources Manager

Part 2 contains an overview of initialization processing and overviews of real storage, expanded storage, virtual storage, and auxiliary storage.

Part 3 describes the members of SYS1.PARMLIB, as well as, processes related to initializing the system, and certain system commands (for example, START GTF and SET IPS). For each member, the book describes the meaning and use of each parameter, syntax rules, syntax examples, value ranges that are syntactically acceptable, default values, and performance notes where applicable. Extended descriptions of related topics are in Parts 4 and 5.

Part 4 describes auxiliary storage management (ASM) with respect to controlling the system's use of page and swap data sets. It describes paging and swapping operations, examines the algorithms used, and explains how these algorithms influence the choice of optimum data set sizes. Examples illustrate typical system tuning problems involving the concepts, algorithms, and data set sizes previously described. Also included are some performance recommendations and answers a series of questions commonly asked about ASM.

Part 5 describes the types of control available through the system resources manager (SRM), the functions used to implement these controls, and concepts for using the SRM parameters. Part 5 also presents guidelines for defining these parameters and describes several sample specifications.

Note: All references to TSO Extensions indicate the program product TSO Extensions (5665-285).

Related Information

Prerequisite Books

To use this book effectively, you need to know about the information in the following books:

Short Title Used	Long Title of the Book	Order Number
MVSCP Guide	MVS/ESA MVS Configuration Program Guide and Reference	GC28-1817
JCL Reference	MVS/ESA JCL Reference	GC28-1829
System Generation	MVS/ESA Installation: System Generation	GC28-1825

Corequisite Books

While you are using this book, you are likely to need the information in the following books:

Short Title Used	Long Title of the Book	Order Number
Application Development Guide	MVS/ESA Application Development Guide	GC28-1821
Application Development Macro Reference	MVS/ESA Application Development Macro Reference	GC28-1822
SPL: Application Development Guide	MVS/ESA SPL: Application Development Guide	GC28-1852
SPL: Application Development Macro Reference	MVS/ESA SPL: Application Development Macro Reference	GC28-1857
System Modifications	MVS/ESA System Modifications	GC28-1831
User Exits	MVS/ESA User Exits	GC28-1836
Conversion Notebook	MVS/ESA Conversion Notebook Volume 1 and Volume 2	GC28-1567 GC28-1568

Other Referenced Books

Where necessary, this book references information in other books using shortened versions of the book title. The following table shows the shortened titles, complete titles, and order numbers.

MVS/ESA Titles

Short Title Used	Long Title of the Book	Order Number
Planning: Dump and Trace Services	Planning: Dump and Trace Services	GC28-1838
TSO Commands	TSO Command Language Reference	
TSO User's Guide	TSO Terminal User's Guide	
R and D Codes	Routing and Descriptor Codes	GC38-1816
IOCP Guide	Input/Output Configuration Program User's Guide and Reference	
TSO/E Customization	TSO Extensions Customization	
JES2 Init and Tuning	System Programming Library: JES2 Initialization and Tuning	
JES3 Init and Tuning	System Programming Library: JES3 Initialization and Tuning	
Service Aids	System Programming Library: Service Aids	GC28-1844
SMF	System Programming Library: System Management Facilities	GC28-1819
JES2 Commands	Operations: JES2 Commands	
System Commands	Operations: System Commands	GC28-1826
	Integrated Catalog Administration: Access Method Service Reference	
VSAM Catlg Adm	VSAM Integrated Catalog Administration: Access Method Service Reference	
System Messages	Message Library: System Messages (2 volumes)	GC28-1812 GC28-1813
Utilities	MVS Data Administration: Utilities	GC26-4018
DFP: Customization	Data Facility Product Version 2: Customization	
Component Diagnosis	Component Diagnosis (multiple volumes)	
TSO/E Commands	TSO Extensions Command Language Reference	
IPCS Command Reference	Interactive Problem Control (IPCS) System Command Reference	GC28-1834
IPCS Planning	Interactive Problem Control (IPCS) Planning and Customization	GC28-1832

Short Title Used	Long Title of the Book	Order Number
IPCS User's Guide.	Interactive Problem Control (IPCS) User's Guide	GC28-1828
Recovery and Reconfig	MVS/Extended Architecture Planning: Recovery and Reconfiguration	GC28-1160

OS/VS2 Titles

Short Title Used	Long Title of the Book	Order Number
Global Resource Serialization	MVS Planning: Global Resource Serialization	GC28-1160
	Mass Storage System Extensions General Information	GH35-0034
	Mass Storage System Extensions Services	SH35-0035

Advanced Communications Function Titles

Short Title Used	Long Title of the Book	Order Number
TCAM Installation	TCAM Version 2 Installation Reference	SC30-3133
VTAM Planning and Installation	VTAM Version 2 Planning and Installation Reference	SC27-0610
VTAM Programming	VTAM Version 2 Programming	SC27-0611

Contents

Figures

Part 1. General Introduction

This manual discusses the following general subjects:

- Storage management overview
- System initialization
- Auxiliary storage manager
- The system resources manager
- The use of GTF to track sysevents

Part 2 - Storage Management Overview

This section describes how storage evolves from the selection of the LOAD function at the system console through initialization, to the point at which several address spaces have been initialized. The discussion presents the following topics:

- The initialization process, which consists of system and storage initialization (including the creation of system component address spaces) and master scheduler initialization (including the initialization of subsystems).

  The initial program load (IPL) and the nucleus initialization program (NIP) perform portions of the initialization process.

- START/LOGON/MOUNT processing.

In addition, the three general types of storage along with their unique areas, are explained. The three general types of storage are as follows:

- Real storage, which includes extended storage
- Virtual storage
- Auxiliary storage

Part 3 - System Initialization

System initialization is the part of system tailoring that takes place after system generation (sysgen), and after running the Input Output Configuration Program (IOCP) and the MVS Configuration Program (MVSCP). The tailoring results from the specification of system parameters, first at initialization and then later when certain operator commands are issued.

System tailoring is the overall process by which an installation selects its operating system. The process consists of specifying system options through these mechanisms:

- System generation
- Initialization-time selections
- Certain operator commands after IPL
- Implicit system parameters

Initialization-time choices that help to tailor the system can come from several sources:

- Various types of IPLs
- Operator entry of parameters from the console

- SYS1.PARMLIB (also referred to as parmlib). This data set is one of the main sources of IPL-time parameters
- The JES2 or JES3 initialization data set

There are several types of IPL:

- *Cold Start*: Any IPL that loads (or reloads) the pageable link pack area (PLPA), but does not preserve virtual I/O (VIO) data set pages. The first IPL after system generation is always a cold start because the PLPA is initially loaded. Subsequent IPLs are cold starts when the PLPA is reloaded either to alter its contents or to restore its contents if they were destroyed.
- *Quick Start*: Any IPL that does not reload the PLPA and does not preserve VIO data set pages. (The nucleus initialization program (NIP) resets the page and segment tables to match the last-created PLPA.)
- *Warm Start*: Any IPL that does not reload the PLPA, but does preserve journaled VIO data set pages.

Operator Entry of Parameters: The operator IPLs the system and responds to NIP's "SPECIFY SYSTEM PARAMETERS" message. The operator's response directs NIP, master scheduler initialization, and other components to the desired parmlib members. The operator enters ENTER to default to the basic general parameter list IEASYS00, or enters SYSP=(aa,bb...) to select one or more alternate general parameter lists, such as IEASYS01, IEASYS02, etc. The alternate lists can supplement or partially override the basic list. The operator need not enter parameter values directly, except for those cases in which parameters are missing, are syntactically invalid, can't be read, or must be supplemented to satisfy a special case. (An example of a special case would be the operator entry of the PAGE parameter to increase the amount of paging space on direct access storage.)

If an error occurs with certain parmlib members, the IPL program prompts the operator to enter manually one or more of the member's parameters. If the parameter can't be corrected, the operator can use ENTER on the console. ENTER causes selection of system defaults if they exist. Most parameters have defaults, either as default parmlib members, or as coded values in system components. If a default doesn't exist, ENTER cancels the parameter. (The defaults are listed in the individual descriptions of parmlib members in "Part 3: System Initialization.")

An operator-entered parameter overrides the same parameter specified in parmlib member IEASYS00 or IEASYSxx, except for:

- A parameter in which operator intervention is prohibited (OPI=NO). In this case, the operator-entered parameter is ignored unless the parmlib parameter is invalid.
- The PAGE parameter. The page data set names entered by the operator are added for the life of the IPL to those specified in either IEASYS00 or IEASYSxx. (For information on the PAGE parameter, refer to the description of member IEASYSxx in Part 3.)

The Use of SYS1.PARMLIB: SYS1.PARMLIB is read by NIP and master scheduler initialization at IPL, and later by such components as the system resources manager, the terminal I/O controller (TIOC), reconfiguration, and the generalized trace facility (GTF), which are invoked by operator command. SYS1.PARMLIB is a single data set containing many initialization parameters in a pre-specified form. The purpose of parmlib is to minimize the need for operator entry of parameters. For example the parameters for the TIOC are described in member IKJPRM00. For GTF, the parameters are under member GTFPARM.

Parmlib contains both a basic or default general parameter list, IEASYS00, and possible alternate general parameter lists, called IEASYSaa, IEASYSbb, etc. Parmlib also contains specified members, such as COMMNDxx, and IEALPAxx. The general parameter list(s) contain both parameter values and "directors." The directors (for example, CMD=4P, MLPA=01) point or direct NIP or master scheduler initialization to one or more specialized members, such as COMMND4P or IEALPA01.

System Tailoring Through Operator Commands: After IPL, several operator commands provide additional system tailoring by directing particular groups of parameters to specific system components. The commands include:

- Stop and start JES2 ($P JES2 and S)
- START tcamproc
- MODIFY tcamproc
- SET IPS, SET ICS, and SET OPT
- SETDMN
- SET MPF
- SETSMF
- SET SMF
- START GTF
- START VLF, SUB=MSTR
- START vtamproc
- SET SLIP
- SET PFK

Part 4 - The Auxiliary Storage Management Initialization

The auxiliary storage manager (ASM) controls paging and swapping. ASM uses algorithms to evenly distribute the I/O load.

This part describes the ways in which page and swap data set sizes can affect system performance and gives examples of space calculation for these data sets.

Part 5 - The System Resources Manager

To a large degree, an installation controls the functioning of the system through the mechanism of the system resources manager (SRM). Part 5 describes the functioning of SRM, the parameters that control its functioning, and guidelines for defining these SRM parameters.

SRM has two principal objectives:

- First, to distribute the system's processing resources among individual address spaces in a way that satisfies the installation's response and turnaround time objectives.
- Second, to attempt to optimize the use of the system's resources by system users (address spaces). This objective is primarily a system throughput consideration.

SRM uses four types of control to accomplish these objectives:

- Domain control
- Workload control
- Resource access control
- Throughput control

These controls are implemented by combinations of the following eight functions:

- Swapping
- Resource access control
- Resource use functions
- Enqueue delay minimization
- I/O priority queueing
- Device allocation
- Prevention of storage shortages
- Pageable frame stealing

Part 2. Storage Management Overview

To tailor the system's storage parameters, you need a general understanding of the system initialization and storage initialization processes. This section contains the following topics:

- The initialization process
- Real storage overview
- Extended storage overview
- Virtual storage overview
- Auxiliary storage overview

Initialization Process

The system initialization process prepares the system control program and its environment to do work for the installation. The process essentially consists of:

- System and storage initialization, including the creation of system component address spaces. The initial program load (IPL) and the nucleus initialization program (NIP) perform this portion of initialization.
- Master scheduler initialization and subsystem initialization.

When the system is initialized and the job entry subsystem is active, the installation can submit jobs for processing by using the START, LOGON, or MOUNT command.

The initialization process begins when the system operator selects the LOAD function at the system console. IPL locates all of the usable, online, real storage that is available to the system, and creates a virtual environment for the building of various system areas.

IPL performs the following major initialization functions:

- Loads the DAT-off nucleus into real storage.
- Loads the DAT-on nucleus into virtual storage so that it spans above and below 16 megabytes (with the exception of the prefixed storage area (PSA), which IPL loads at virtual zero).
- Builds the nucleus map, NUCMAP, of the DAT-on nucleus. NUCMAP resides in virtual storage above the nucleus.
- Builds the page frame table (PFT) in virtual storage above the NUCMAP.
- Allocates the system's minimum virtual storage for the system queue area (SQA) and the extended SQA.
- Allocates virtual storage for the extended local system queue area (extended LSQA) for the master scheduler address space.

When IPL processing completes, NIP continues to initialize the system by interpreting and acting upon the system parameters that were specified. NIP performs the following major initialization functions:

- Expands the SQA and the extended SQA by the amounts specified on the SQA system parameter.
- Creates the pageable link pack area (PLPA) and the extended PLPA for a cold start IPL; resets tables to match an existing PLPA and extended PLPA for a quick start or a warm start IPL.
- Loads modules into the fixed link pack area (FLPA) or the extended FLPA. Note that NIP performs this function only if the FIX system parameter is specified.
- Loads modules into the modified link pack area (MLPA) and the extended MLPA. Note that NIP performs this function only if the MLPA system parameter is specified.
- Allocates virtual storage for the common service area (CSA) and the extended CSA. The amount of storage allocated depends on the values specified on the CSA system parameter at IPL.
- Page protects the: NUCMAP, PLPA and extended PLPA, MLPA and extended MLPA, FLPA and extended FLPA, and portions of the nucleus.

  Note: An installation can override page protection of the MLPA and FLPA by specifying NOPROT on the MLPA and FIX system parameters.

See Figure 2-1 for the relative position of the system areas in virtual storage. Most of the system areas exist both below and above 16 megabytes providing an environment that can support both 24-bit and 31-bit addressing. However, each area and its counterpart above 16 megabytes can be thought of as a single logical area in virtual storage.

System Address Space Creation

In addition to initializing system areas, NIP establishes system component address spaces. NIP establishes an address space for the master scheduler (the master scheduler address space) and other system address spaces for various subsystems and system components. Some of the component address spaces are:

- Program call/authorization for cross-memory communications
- System trace
- Global resource serialization
- Dumping services

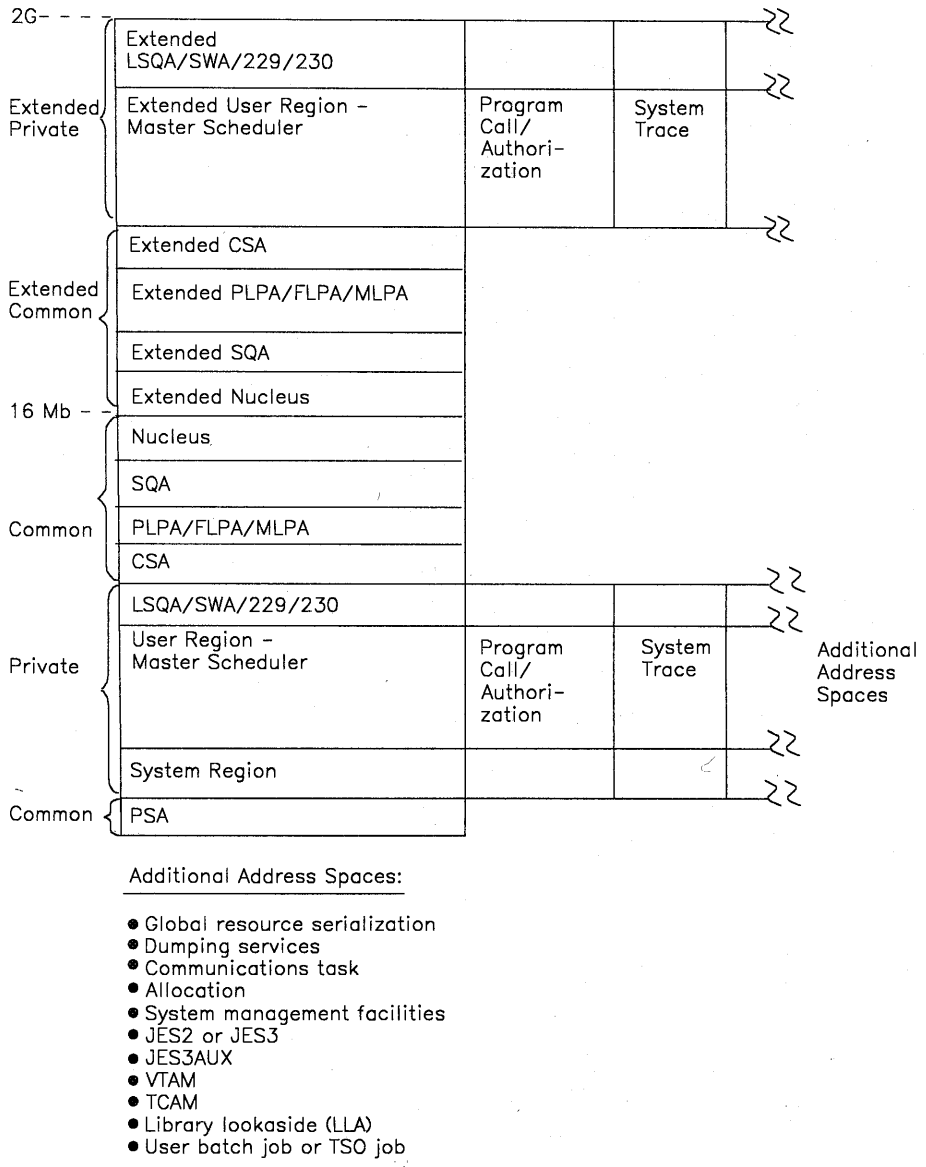


Figure 2-1. Virtual Storage Layout for Multiple Address Spaces

Master Scheduler Initialization

Master scheduler initialization routines initialize system services such as the system log and communications task, and start the master scheduler itself. They also cause creation of the system address space for the job entry subsystem (JES2 or JES3), and then start the job entry subsystem.

Note: When JES3 is the primary job entry subsystem, a second JES3 address space (JES3AUX) can be optionally initialized after master scheduler initialization completes. The JES3AUX address space is an auxiliary address space that contains JES3 control blocks and data.

Subsystem Initialization

Subsystem initialization is the process of readying a subsystem for use in the system; this involves defining the subsystem's name (the name it will be known by) and processing the subsystem so that the system recognizes it by name. IEFSSNxx members of SYS1.PARMLIB contain the definitions for the primary subsystems, such as JES2 or JES3, and the secondary subsystems, such as VPSS and DB2.

During system initialization, the defined subsystems are initialized. Define the primary subsystem (JES) first. Most subsystems, such as DB2, require the services of the primary subsystem in their initialization routines. Problems can occur if subsystems that use the subsystem affinity service in their initialization routines are initialized before the primary subsystem. After the primary JES is initialized, then the subsystems are initialized in the order in which the IEFSSNxx parmlib members are specified by the SSN=parameter. For example, for SSN=(aa,bb) parmlib member IEFSSNaa would be processed before IEFSSNbb.

Using IEFSSNxx to initialize the subsystems, you can specify the name of a subsystem initialization routine to be given control during master scheduler initialization, and you can specify the input parameter to be passed to the subsystem initialization routine. IEFSSNxx is described in more detail in "Part 3: System Initialization."

Note: The Storage Management Subsystem (SMS) is the only subsystem that can be defined before the primary subsystem. Refer to the description of parmlib member IEFSSNxx in "Part 3: System Initialization" for SMS considerations.

START/LOGON/MOUNT Processing

After the system is initialized and the job entry subsystem is active, jobs may be submitted for processing. When a job is activated via START (for batch jobs), LOGON (for time-sharing jobs) or MOUNT, a new address space must be allocated. Note that prior to LOGON, the operator must have started TCAM or VTAM/TCAS, which have their own address spaces. Figure 2-1 is a virtual storage map containing or naming the basic system component address spaces, the optional TCAM and VTAM system address spaces, and a user address space.

The system resources manager decides, based on resource availability, whether a new address space can be created. If not, the new address space will not be created until the system resources manager finds conditions suitable.

Real Storage Overview

This section provides an overview of real storage. For a discussion of extended storage, see "Extended Storage Overview."

The system uses a portion of both real storage and virtual storage. In order to determine how much real storage is available to the installation, the system's fixed storage requirements must be subtracted from the total real storage. The real storage available to an installation can be used for the concurrent execution of the paged-in portions of any installation programs.

The real storage manager (RSM) controls the allocation of real storage during initialization and pages in user or system functions for execution. Some RSM functions:

- Allocate real storage to satisfy GETMAIN requests for SQA and LSQA.

- Allocate real storage for page fixing.
- Allocate real storage for an address space that is to be swapped in.
- Allocate and initialize control blocks and queues related to extended storage.

If there is storage above 16 megabytes, RSM allocates real storage locations above 16 megabytes for SQA, LSQA, and the pageable requirements of the system. When non-fixed pages are fixed for the first time, RSM:

- Ensures that the pages occupy the appropriate type of frame
- Fixes the pages and records the type of frame used

In subsequent real storage allocations for such pages, RSM will allocate frames based on the previous real storage requirements for the pages. Thus, RSM minimizes the movement of pages from locations above 16 megabytes to locations below 16 megabytes.

Pages that must reside in real storage below 16 megabytes include:

- SQA subpool 226 pages.
- Fixed pages obtained using the RC, RU, VRC, or VRU form of GETMAIN and one of the following is true:
  - LOC=BELOW is specified.
  - The default, LOC=RES, is either specified or taken, and the program issuing the GETMAIN resides below 16 megabytes virtual.
- Fixed pages obtained using the LU, LC, EU, EC, VU, VC, or R form of GETMAIN.
- V=R pages.

Pages that can reside in real storage above 16 megabytes include:

- Nucleus pages.
- SQA subpools 239 and 245 pages.
- LSQA pages.
- All pages with virtual addresses greater than 16 megabytes.
- Fixed pages obtained using the RC, RU, VRC, or VRU form of GETMAIN and one of the following is true:
  - LOC=(BELOW,ANY) is specified.
  - LOC=(RES,ANY) is specified.
  - LOC=ANY is specified.
  - LOC=(ANY,ANY) is specified.
  - The default, LOC=RES, is either specified or taken, and the program issuing the GETMAIN resides above 16 megabytes virtual.
- Any non-fixed page.

Each installation is responsible for establishing many of the real storage parameters that govern RSM's processing. The following overview describes the function of each area composing real storage.

The primary requirements/areas composing real storage are:

1. The basic system fixed storage requirements — the nucleus, the allocated portion of SQA, and the fixed portion of CSA.
2. The private area fixed requirements of each swapped-in address space — the LSQA for each address space and the page-fixed portion of each virtual address space.

Once initialized, the basic system fixed requirements (sometimes referred to as global system requirements) remain the same until system parameters are changed. Fixed storage requirements (or usage) will, however, increase as various batch or time sharing users are swapped-in. Thus, to calculate the approximate fixed storage requirements for an installation, the fixed requirements for each swapped-in address space must be added to the basic fixed system requirements. Fixed requirements for each virtual address space include system storage requirements for the LSQA (which is fixed when users are swapped in) and the real storage estimates for the page-fixed portions of the installation's programs.

The real storage for the processor(s), reduced by the global fixed and paged-in virtual storage required to support installation options, identifies the real storage remaining to support swapped-in address spaces. The total number of jobs that can be swapped in concurrently can be determined by estimating the working set (the percentage of virtual storage that must be paged in for the program to run effectively) for each installation program. The working set requirements will vary from program to program and will also change dynamically during execution of the program. Allowances should be made for *maximum* requirements when making the estimates.

System Preferred Area

To enable a V=R allocation to occur and storage to be varied offline, RSM performs special handling for the following types of pages:

- SQA
- LSQA for non-swappable address spaces
- Fixed page frame assignments for non-swappable address spaces

Because RSM cannot, upon demand, free the frames used for these page types, real storage could become fragmented (by the frames that could not be freed.) Such fragmentation could prevent a V=R allocation or prevent a storage unit from being varied offline. Therefore, for all storage requests for the types of pages noted, RSM allocates storage from the preferred area to prevent fragmentation of the nonpreferred "reconfigurable" area.

A system parameter, RSU, allows the installation to specify the number of storage units that are to be kept free of long-term fixed storage allocations, and thus be available for varying offline. Once this limit is established, the remainder of real storage, excluding storage reserved for V=R allocation, is marked as preferred area storage and used for long-term fixed storage allocation.

Nucleus Area

The nucleus area contains the nucleus load module and extensions to the nucleus that are initialized during IPL processing.

The DAT-on nucleus is loaded into virtual storage so that it spans above and below 16 megabytes. The DAT-on nucleus is fixed for the duration of each IPL. Although its size varies depending on an installation's configuration, it doesn't change as additional jobs are swapped in and out.

The DAT-off nucleus is loaded into contiguous high real storage and is not mapped virtually. The DAT-off nucleus is also fixed for the duration of an IPL.

The Fixed Link Pack Area (FLPA)

An installation can elect to have some modules that are normally loaded in the pageable link pack area (PLPA) loaded into the fixed link pack area (FLPA). This area should be used only for modules that significantly increase performance when they are fixed rather than pageable. Modules placed in the FLPA must be reentrant and refreshable.

The FLPA exists only for the duration of an IPL. Therefore, if an FLPA is desired, the modules in the FLPA must be specified for each IPL (including quick-start and warm-start IPLs).

It is the responsibility of the installation to determine which modules, if any, to place in the FLPA. Note that if a module is heavily used and is in the PLPA, the system's paging algorithms will tend to keep that module in real storage. The best candidates for the FLPA are modules that are infrequently used but are needed for fast response to some terminal-oriented action.

Specified by: A list of modules to be put in FLPA must be established by the installation in the fixed LPA list (IEAFIXxx) member of SYS1.PARMLIB. Modules from the LPALST concatenation, the LNKLST concatenation, SYS1.MIGLIB, and SYS1.SVCLIB can be included in the FLPA. FLPA is selected with the RESIDNT parameter of the DATASET system generation macro instruction, through specification of the FIX system parameter in IEASYSxx, or from the operator's console at system initialization.

System Queue Area (SQA-Fixed)

SQA is allocated in fixed storage upon demand as long-term fixed storage and remains so until explicitly freed. The number of real frames assigned to SQA may increase and decrease to meet the demands of the system.

All SQA requirements are allocated in 4K frames as needed. These frames are placed within the preferred area (above 16 megabytes, if possible) to keep long-term resident pages grouped together.

If no space is available within the preferred area, and none can be obtained by stealing a non-fixed/unchanged page, then the "reconfigurable area" is reduced by one storage band, and the band is marked as preferred area storage. A band is the basic unit of physical storage. If there is no "reconfigurable area" to be reduced, a page is assigned from the V=R area. Excluded from page stealing are frames that have been fixed (for example, via the PGFIX macro instruction), allocated to a V=R region, placed offline using a CONFIG command, have been changed, have I/O in progress, or contain a storage error.

Fixed LSQA Storage Requirements

Except for the extended private area page tables, which are pageable, the local system queue area (LSQA) for any swapped-in address space is fixed in real storage (above 16 megabytes, if possible). It remains so until it is explicitly freed or until the end of the job step or task associated with it. The number of LSQA frames allocated in real storage might increase or decrease to meet the demands of the system. If preferred storage is required for LSQA and no space is available in the preferred area, and none can be obtained by stealing a non-fixed/unchanged page, then the "reconfigurable area" is reduced by one storage band, and the band is marked as preferred area storage. If there is no "reconfigurable area" to be reduced, a page is assigned from the V=R area.

V=R Real Storage Space

This area is used for the execution of job steps specified as fixed by virtue of being assigned to V=R regions in virtual storage (see "Real Regions" under the "Virtual Storage Overview"). Such jobs run as nonpageable and nonswappable.

The V=R area is allocated starting directly above the system region in real storage. The virtual addresses for V=R regions are mapped one-to-one with the real addresses in this area. When a job requests a V=R region, the lowest available address in the V=R area in real storage, followed by a contiguous area equal in size to that of the V=R region in virtual storage, is located and allocated to the region.

If there is not enough V=R space available in the V=R area, the allocation and execution of new V=R regions are prohibited until enough contiguous storage is made available.

The V=R area can become fragmented because of system allocation for SQA and LSQA or because of long-term fixing. When this happens, it becomes more difficult — and may be impossible — for the system to find contiguous storage space for allocating V=R regions. Such fragmentation may last for the duration of an IPL. It is possible that fragmentation will have a cumulative effect as long-term fixed pages are occasionally assigned frames from the V=R area.

Specified by:

- the REAL parameter of the IEASYSxx member
- use of the REAL parameter from the operator's console during NIP

Extended Storage Overview

Note: Extended storage and expanded storage are terms that have the same meaning. This book uses the term extended storage.

Extended storage can be thought of as an extension of real storage. RSM manages extended storage in a way that is similar to how it manages real storage. The purpose of extended storage is to reduce the paging and swapping of pages between real storage and auxiliary storage, and thus enhance system performance. Because moving a page between real storage and extended storage is a synchronous operation, use of extended storage can provide a significant performance advantage.

RSM uses extended storage as an extension of real storage. When a page is to be removed from real storage, RSM first considers moving it to extended storage instead of auxiliary storage. When a page that is needed is not in real storage, RSM first checks extended storage for the page. If the page is on extended storage, RSM synchronously retrieves the page. If the page is not on extended storage, RSM calls ASM to schedule asynchronously the paging I/O to retrieve the page from auxiliary storage. For a more detailed explanation of how pages are sent to extended storage, see the *Extended storage control* topic in "Part 5. The System Resources Manager."

When contention for extended storage increases, the system removes pages from extended storage to free extended storage frames. RSM first moves the pages from extended storage to real storage. RSM then calls ASM to schedule the paging I/O necessary to send these pages to auxiliary storage. This process is called **migration**. Migration completes when these pages are actually sent to auxiliary storage.

Virtual I/O in Extended Storage: Extended storage can be used for virtual I/O (VIO) pages for:

- Any VIO job in a system with no VIO journaling, or
- A non-journaled VIO job regardless of whether VIO journaling is active on the system.

Extended storage is only used for VIO when no significant increase in demand or swap paging will result. The system supplied VIO criteria age default value of 900 seconds prevents directing VIO pages to extended storage before all other categories of pages. The criteria age value may be changed by modifying the IEAOPTxx member of SYS1.PARMLIB (see IEAOPTxx in Chapter 5).

The storage isolation function of SRM can be used to override the criteria table. Because storage isolation considers VIO pages created by an address space to be part of the address space working set, sufficient amounts of real and expanded storage can be made available to keep VIO data in expanded storage. For example, if an application that requires 1 megabyte of processor storage to execute without paging, creates 100 megabytes of temporary data that is kept in a dataset assigned to VIO, then a performance group can be set up for the job and the minimum processor storage allocation can be set to 101 megabytes. This allows the VIO data to be maintained in expanded storage, regardless of the migration age.

The storage isolation function can also be used to limit the maximum amount of processor storage for certain applications when there is contention for real and expanded storage.

If storage isolation is used to set a minimum or maximum working set size for an application that uses VIO datasets, the following considerations should be made:

- Since VIO data is included in the working set, the number of VIO pages that will be generated should be considered when calculating the minimum working set size.
- An application that has more real and expanded than the maximum working set is a preferred source for page replacement regardless of the frequency of reference to the data. The maximum working set should only be used when this type of control is required.

Notes:

1. When there is not sufficient expanded storage, the VIO activity will go to the paging subsystem. Therefore, the paging subsystem should be configured to handle the load that could result.
2. VIO to extended storage is not selective. If VIO is allowed on a system then any application may choose to use it. User exit routines must be used if it is necessary to restrict the VIO to certain applications or control space allocation.

Swapping and Migration

RSM is responsible for reclaiming the real storage allocated to an address space when it is to be swapped out of real storage. RSM is also responsible for building the control structures necessary to efficiently swap the address space back into real storage. When an address space is swapped out of real storage, RSM works with SRM to identify the working set pages that will be swapped back into real storage.

If the address space is swapped directly from real storage to auxiliary storage, ASM reads and writes these working set pages in parallel. This is called a **single-stage swap-in**

If the address space is swapped from real storage to extended storage, RSM and SRM divide the working set pages into a primary working set and a secondary working set. The primary working set consists of LSQA pages, fixed pages, and one page from each virtual storage segment that is included in the working set. RSM manages the primary working set as one entity. The secondary working set consists of all working set pages not included in the primary working set.

If the address space is swapped back into real storage from extended storage, RSM must return the entire primary working set before the address space can become dispatchable. If the address space is migrated from extended storage to auxiliary storage, RSM migrates the secondary working set in groups based on the need to replenish the extended storage available frame queue. Auxiliary storage management (ASM) then writes these secondary working set pages as group requests to the page data sets.

When the entire secondary working set has been selected for migration, RSM swaps the primary working set into real storage, and then out to auxiliary storage. When an address space that has been migrated is swapped in, RSM first swaps in the primary working set, and then swaps in the secondary working set. This process is called a **two-stage swap-in**.

Virtual Storage Overview

Estimating the virtual storage allocated at an installation is important primarily because this storage must be backed up by real storage in some ratio (for example, 25%). This backup storage contributes significantly to an installation's total real storage requirements.

Virtual storage must also be backed up by auxiliary storage. See the discussion of paging data space in "Part 4. Auxiliary Storage Management Initialization."

Each installation can use virtual storage parameters to specify how certain virtual storage areas are to be allocated. These parameters have an impact on real storage use and overall system performance. The following overview describes the function of each virtual storage area.

The Virtual Storage Address Space

A two-gigabyte virtual storage address space is provided for:

- The master scheduler address space
- JES
- Other system component address spaces, such as allocation, system trace, system management facilities, and dumping services
- Each user (batch or time-shared)

The system uses a portion of each virtual address space in which a user's job or job step (whether batch or time shared) logically resides.

Each virtual address space consists of:

- The ***common area(s)*** below 16 megabytes
- The ***private area*** below 16 megabytes
- The ***extended common area*** above 16 megabytes
- The ***extended private area*** above 16 megabytes

Figure 2-2 shows the layout of the storage areas for an address space in virtual storage. Note that most of the system areas exist both below and above 16 megabytes providing an environment that can support both 24-bit and 31-bit addressing. However, each area and its counterpart above 16 megabytes can be thought of as a single logical area in virtual storage.

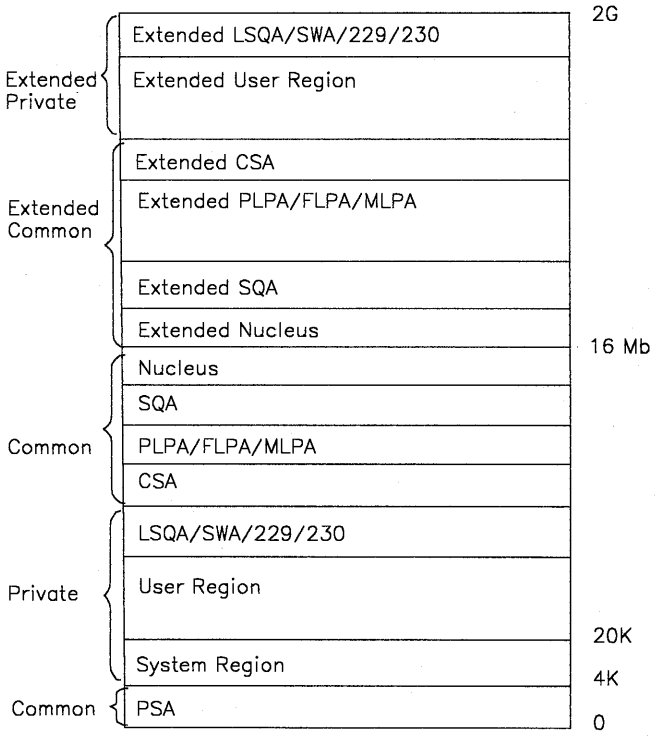


Figure 2-2. Virtual Storage Layout for a Single Address Space

The ***common area*** contains system control programs and control blocks. The following storage areas are located in the common area:

- Prefixed storage area (PSA). (There is no PSA in the extended common area.)
- Common service area (CSA).
- Pageable link pack area (PLPA).
- Fixed link pack area (FLPA).
- Modified link pack area (MLPA).
- System queue area (SQA).
- Nucleus, which is fixed and nonswappable.

Except as noted for the PSA, each storage area in the common area (below 16 megabytes) has a counterpart in the extended common area (above 16 megabytes).

Each address space uses the same common area. Portions of the common area are paged in and out as the demands of the system change and as new user jobs (batch or time-shared) start and old ones terminate.

The ***private area*** contains:

- A local system queue area (LSQA).
- A scheduler work area (SWA).
- Subpools 229/230 (the authorized user key area).
- A 16K system region area. (This area does not exist in the extended private area.)
- Either a V=V (virtual = virtual) or V=R (virtual = real) private user region for running programs and storing data.

Except for the 16K system region area, each storage area in the private area (below 16 megabytes) has a counterpart in the extended private area (above 16 megabytes).

Each address space has its own unique private area allocation. The private area (except LSQA) is pageable unless a user specifies a V=R region. If assigned as V=R, the actual V=R region area (excluding SWA, subpools 229/230, and the 16K system region area) is fixed and nonswappable.

General Virtual Storage Allocation Considerations

Virtual storage allocated in each address space is divided between the system's requirements and the user's requirements. The basic system control programs require space from each of the basic areas; supervisor routines operate primarily from the fixed and common areas, and the scheduling routines operate in the private and common areas.

Storage for SQA, CSA, LSQA, and SWA is assigned in behalf of either the system or a specific user. Generally, space is assigned the system in SQA and CSA and for users in LSQA or SWA.

System Queue Area (SQA/Extended SQA)

This area contains tables and queues relating to the entire system. Its contents are highly dependent on configuration and job requirements at an installation. The total amount of virtual storage, number of private virtual storage address spaces and size of the installation performance specification table are some of the factors that affect the system's use of SQA.

The SQA is allocated directly below the nucleus; the extended SQA is allocated directly above the extended nucleus.

The size of the SQA can be specified by:

- through the SQA parameter in the IEASYS00 member of SYS1.PARMLIB
- through the NIP or operator's console

If the specified amount of virtual storage is not available during initialization, a warning message will be issued. The SQA parameter may be respecified at that time from the operator's console.

Virtual SQA is allocated as a number of 64K blocks to be added to the minimum system requirements for SQA. If the SQA required by the system configuration exceeds the amount that has been reserved via the SQA parameter, the system attempts to allocate additional virtual SQA from the CSA area. If less than 8 pages (below 16 megabytes) remain for allocation purposes, the creation of new address spaces is halted. When SQA is in use, it is fixed in real storage.

The size of the SQA cannot be increased or decreased during a restart that reuses the previously initialized PLPA (a quick start). The size will be the same as during the preceding IPL.

Note: If the size allocated for extended SQA is too small or is used up very quickly, the system attempts to steal pages from extended CSA. When both extended SQA and extended CSA are used up, the system allocates space from SQA and CSA below the 16 megabyte line. The allocation of this storage could eventually lead to a system failure. Ensuring the appropriate size of extended SQA and extended CSA storage is critical to the long-term operation of the system.

Pageable Link Pack Area (PLPA/Extended PLPA)

This area contains SVC routines, access methods, and other read-only system programs along with any read-only reenterable user programs selected by an installation that can be shared among users of the system. Optional functions or devices selected by an installation during sysgen add additional modules to the LPALST concatenation.

It is desirable to place all frequently used refreshable SYS1.LINKLIB and SYS1.CMDLIB modules[1] in the PLPA because of the following advantages. Any installation may also specify that some reenterable modules from the LNKLST concatenation, SYS1.SVCLIB, and/or the LPALST concatenation be placed in a fixed extension to the link pack area (FLPA) to further improve performance (see "Fixed Link Pack Area" under "Real Storage Overview").

- If possible, PLPA is backed by real storage above 16 megabytes; real storage below 16 megabytes is available for other uses.
- The length of time that a page occupies real storage depends on its frequency of use. If the page is not used over a period of time, the system will reuse (steal) the real storage frame that the page occupies.
- The most frequently used PLPA modules in any given time period will tend to remain in real storage.
- PLPA paged-in modules avoid program fetch overhead.
- Two or more programs that need the same PLPA module share the common PLPA code, thus reducing the demand for real storage.
- The main cost of unused PLPA modules is paging space, because only auxiliary storage is involved when modules are not being used.
- All modules in the PLPA are treated as refreshable, and are not paged-out. This action reduces the overall paging rate compared with modules in other libraries.

Modules loaded into the PLPA are packed within page boundaries. Modules larger than 4K begin on a page boundary with smaller modules filling out. PLPA can be used more efficiently through use of the LPA packing list (IEAPAKxx). IEAPAKxx allows an installation to pack groups of related modules together, which can sharply reduce page faults. The total size of modules within a group should not exceed 4K, and the residence mode (RMODE) of the modules in a group should be the same. (See the description of the IEAPAKxx parmlib member in "Part 3: System Initialization.")

System Performance: The following recommendations should improve system performance:

- Copy frequently and moderately used refreshable modules from the LNKLST concatenation and SYS1.CMDLIB to the LPALST concatenation. Avoid duplicating modules, module names, or alias names that already exist in the LPALST concatenation.

---

[1] See an alternative suggestion on the placement of some PLPA modules and SYS1.CMDLIB modules later in this topic.

Note: Maintain the following EDIT subcommands in the SYS1.CMDLIB: ALLOCATE, ATTRIB, EXEC, FREE, HELP, PROFILE, RUN, SEND, and SUBMIT.

When a page in the PLPA is not continually referred to by multiple address spaces, it tends to be stolen. The reason is that individual address spaces typically get swapped out (without the PLPA pages to which they refer) and, as a result, one address space can't maintain the referencing frequency necessary to prevent page stealing.

In contrast, low-use TSO/E command processors and low-use user programs probably should not be placed in the LPALST concatenation. Allowing low-use SYS1.CMDLIB and low-use user modules to remain in their libraries and be fetched into the user's subpools, instead of placing them in the PLPA, causes the TSO/E command processor and user programs to be swapped in and out with the address space.

- Ideally, place the PLPA on a single page data set on a single device. This happens only if the first-named page data set (specified at IPL) is large enough to hold the entire PLPA. If it is not large enough, the PLPA overflows to the common page data set. Because the LPA directory is paged out with the last 100K block, such PLPA separation could degrade performance, particularly if the over-flow device is slower than the primary device.
- Minimize page faults and disk arm movement by specifying module packing through the IEAPAKxx member of parmlib. Module packing reduces page faults by placing in the same virtual page those small modules (less than 4K bytes) that refer to each other frequently. In addition, module groups that frequently refer to each other but that exceed 4K bytes in combined size can be placed in adjacent (4K) auxiliary storage slots to reduce seek time. Thus, use of IEAPAKxx should improve performance compared with the simple loading of the PLPA from the LPALST concatenation. (See the description of parmlib member IEAPAKxx in "Part 3: System Initialization.")
- Use those parmlib lists that speed the search for listed modules at the expense of the unlisted modules. The lists include the modified LPA list (IEALPAxx) and the fixed LPA list (IEAFIXxx). (For information on these lists, see IEALPAxx and IEAFIXxx in "Part 3: System Initialization.") NIP builds and chains contents directory entries (CDEs) for modules in these lists, one CDE per module.

When a module is requested, the program manager makes a serial search for the requested module (requested in EP[2] form without DCB) in the following sequence:

1. Job pack area (JPA) queue
2. Tasklib(s), steplib, joblib
3. Active LPA queue (via a serial search). This queue contains CDEs for:
   a. fixed LPA (FLPA) modules, as specified in IEAFIXxx members
   b. modified LPA (MLPA) modules, as specified in IEALPAxx members

   Note: Some subsystems, such as IMS, add CDEs to this queue.

---

2 EP means the entry point parameter of an ATTACH, LINK, LOAD, or XCTL macro.

4. Pageable LPA
5. SYS1.LINKLIB and libraries concatenated to it via LNKLSTxx

   Note: Be sure to concatenate SYS1.MIGLIB immediately after SYS1.LINKLIB.

Modules in the modified LPA list and the fixed LPA list tend to be found fairly rapidly, but at the expense of modules that *must* be found in the LPA directory. It's undesirable to make these parmlib lists excessively long because the CDE search will be prolonged and the LPA directory search will be delayed.

- Use the FLPA to reduce page-fault overhead and increase performance at the expense of some of the real storage. Such a trade-off is desirable with a system that tends to be CPU bound. Therefore, it is desirable to divert some of the real storage from possible use by additional address spaces, and use the diverted storage for additional LPA modules. Implement the trade-off by placing moderately used modules that are in the LPALST concatenation into the FLPA (via parmlib member IEAFIXxx). High usage PLPA modules probably need not be listed in IEAFIXxx, because they might be referenced frequently enough to remain in real storage anyway. A large FLPA means that less real storage is available for pageable programs. Accordingly, the system resource manager maintains fewer address spaces in real storage than would otherwise be the case. No loss in throughput should occur, however, as long as CPU use remains reasonably high.

  Note: In real storage, the FLPA will be backed below 16 megabytes. The extended FLPA will be backed above 16 megabytes, if possible.

Modified Link Pack Area (MLPA/Extended MLPA)

This area may be used to contain reenterable routines from SYS1.SVCLIB, the LNKLST concatenation, or the LPALST concatenation that are to be part of the pageable extension to the link pack area during the current IPL. The MLPA exists only for the duration of an IPL. Therefore, if an MLPA is desired, the modules in the MLPA must be specified for each IPL (including quick start and warm start IPLs).

The MLPA is allocated just below the FLPA (or the PLPA, if there is no FLPA); the extended MLPA is allocated above the extended FLPA (or the extended PLPA if there is no extended FLPA). When the system searches for a routine, the MLPA is searched before the PLPA.

The MLPA can be used to temporarily modify or update the PLPA with new or replacement modules. No actual modification is made to the quick start PLPA stored in the system's paging data sets. The MLPA is read-only, unless NOPROT is specified on the MLPA system parameter.

Specified by:

- including a module list as an IEALPAxx member of SYS1.PARMLIB; where xx is the specific list. A list is placed in this member by using the IEBUPDTE system utility prior to an IPL that will use it.
- specification of the MLPA system parameter in IEASYSxx or from the operator's console during system initialization.

Common Service Area (CSA/Extended CSA)

This area contains pageable and fixed system data areas that are addressable by all active virtual storage address spaces. CSA normally contains data referenced by a number of system address spaces, enabling address spaces to communicate by referencing the same piece of CSA data.

CSA is allocated directly below the MLPA; extended CSA is allocated directly above the extended MLPA. If the virtual SQA space is depleted, the system will allocate additional SQA pages from the CSA.

Specified by:

- The SYSP parameter at the operator's console to specify an alternative system parameter list (IEASYSxx) that contains a CSA specification. This alternative list can be created using the IEBUPDTE system utility.
- The CSA parameter at the operator's console during system initialization. This value overrides the current system parameter value for CSA that was established by IEASYS00 or IEASYSxx.

Note: If the size allocated for extended SQA is too small or is used up very quickly, the system attempts to steal pages from extended CSA. When both extended SQA and extended CSA are used up, the system allocates space from SQA and CSA below the 16 megabyte line. The allocation of this storage could eventually lead to a system failure. Ensuring the appropriate size of extended SQA and extended CSA storage is critical to the long-term operation of the system.

Local System Queue Area (LSQA/Extended LSQA)

Each virtual address space has an LSQA. The area contains tables and queues associated with the user's address space. It includes page tables and control blocks needed for all tasks in the address space. The size of the LSQA for any address space depends upon system requirements, such as the number of outstanding timer requests, the number of terminal attentions, and the number of tasks.

LSQA is intermixed with SWA and subpools 229/230 downward from the bottom of the CSA into the unallocated portion of the private area, as needed. Extended LSQA is intermixed with SWA and subpools 229/230 downward from the 2-gigabyte line into the unallocated portion of the extended private area, as needed. (See Figure 2-2.) LSQA will not be taken from space below the top of the highest storage currently allocated to the private area user region. Any job will abnormally terminate unless there is enough space for allocating LSQA.

Scheduler Work Area (SWA/Extended SWA)

This area contains control blocks that exist from task initiation to task termination. It includes control blocks and tables created during JCL interpretation and used by the initiator during job step scheduling. Each initiator has its own SWA within the user's private area.

To enable recovery, the SWA can be recorded on direct access storage in the JOB JOURNAL. SWA is allocated at the top of each private area intermixed with LSQA and subpools 229/230.

Subpools 229/230 - Extended 229/230

This area allows local storage within a virtual address space to be obtained in the requestor's storage protect key. The area is used for control blocks (those placed by system components on behalf of the user, such as DEBs) that can be obtained only by authorized programs having appropriate storage protect keys.

These subpools are intermixed with LSQA and SWA. Subpool 229 is fetch protected, subpool 230 is not. Both subpools are pageable, and both are freed automatically at task termination.

System Region

This area is reserved for GETMAINs by all system functions (for example, ERPs) running under the region control tasks. It comprises 16K (except in the master scheduler address space, in which it has a 200K maximum) of each private area immediately above the PSA. V=V region space allocated to user jobs is allocated upwards from the top of this area. This area is pageable and exists for the life of each address space.

The Private Area User Region/Extended Private Area User Region

The portion of the user's private area within each virtual address space that is available to the user's problem programs is referred to as the user region. It may be any size up to the size of the entire private area minus the size of LSQA/SWA/the user key area (subpools 229 and 230) and the system region (see Figure 2-2).

There are two basic types of regions: virtual and real. Virtual and real regions are mutually exclusive; any private areas can be assigned to V=R or V=V but not both. It is an installation's responsibility to determine which jobs should be assigned to virtual or real regions. In general, V=R should be assigned to regions containing jobs that cannot run in the V=V environment or that are not readily adaptable to it. Programs requiring a one to one mapping from virtual to real storage (for example, PCI driven channel programs) may be candidates for V=R regions.

Two significant differences between these region types is their effect on real storage requirements for an installation and the address translation performed on them. Virtual regions are pageable and swappable. Only enough real storage need be allocated to store the paged-in portion of the job (plus its LSQA). The virtual address of programs running from V=V regions are translated to locate their real storage equivalent. For real regions, real storage for the entire user region is allocated in virtual storage at the time the region is allocated and fixed in real storage.

Virtual Regions

V=V regions begin at the top of the system region (see Figure 2-2) and are allocated upward to the bottom of the area containing LSQA, SWA, and the user key area (subpools 229/230). As a portion of any V=V job is paged in, 4K multiples of real storage are allocated from the available pages in the reserved storage area. Real storage is dynamically assigned and freed on a demand basis as the job executes. V=V region requests that specify a specific region start address are supported only for restart requests, and must specify an explicit size.

For the V=V regions, an individual job's region size is limited by the IEALIMIT routine or the IEFUSI user exit. Any attempt to use more storage than the limit results in an abnormal termination of the job step.

IBM-supplied defaults that limit the user region sizes below and above 16 megabytes are the same regardless of whether the installation uses the IEALIMIT routine or the IEFUSI user exit. The default limit for the user region size below 16 megabytes equals (1) the region size that is specified in the JCL, plus 64K, or (2) the JES default, plus 64K, if no region size is specified in the JCL. The default limit for the user region size above 16 megabytes is 32 megabytes.

Note: If the region size specified in the JCL is zero, the region will be limited by the size of the private area.

The installation can change the limit for the region size below 16 megabytes using IEALIMIT or IEFUSI. However, to change the limit for the region size above 16 megabytes, the installation must use the IEFUSI user exit. See *SPL: User Exits* for information on IEALIMIT and IEFUSI. *SPL: System Modifications* also contains information on these exits.

Real Regions

V=R regions are assigned a virtual space within the private area which maps one for one with the real addresses in real storage below 16 megabytes. Real storage for the entire region is allocated and fixed when the region is first created. In fact, real storage is allocated before its virtual counterpart to increase the possibility of allocating a contiguous space in real storage of sufficient size for a V=R region's requirements.

Specified by: A user can specify that each job (or job step) be assigned in a virtual or real private region area by use of the ADDRSPC parameter of the JOB or EXEC statement. Any explicitly specified ADDRSPC parameter on a JOB statement overrides all EXEC statement ADDRSPC parameters. However, if the ADDRSPC parameter on a JOB statement is defaulted to virtual, then any EXEC having ADDRSPC=REAL specified will be recognized and that job step will have a real region; any steps for which the ADDRSPC parameter is not specified will be assigned as virtual regions.

ADDRSPC is used in conjunction with the REGION parameter. For V=R jobs, the REGION parameter specifies a limit to the size of the V=R region within the private area. This value cannot be greater than the value of the REAL system parameter specified for the IPL.

If a real region's size is not explicitly specified, the VRREGN system parameter is the default size. VRREGN should not be confused with the REAL system parameter. REAL specifies the total amount of real storage that is to be reserved for running all active V=R regions. VRREGN specifies the default subset of that total space that is required for an individual job that does not have a region size specified in its JCL.

A default VRREGN parameter exists in the IEASYS00 member of SYS1.PARMLIB. An installation can change this default by:

- specifying a VRREGN parameter in an alternate system parameter list IEASYSxx. This alternate list can be created by using the IEBUPDTE system utility.
- using the VRREGN parameter at the operator's console during system initialization. This value will then override any current system value for VRREGN that was established by IEASYS00 or IEASYSxx.

Or the installation can override the default by using the REGION parameter in the JOB or EXEC statement.

All region sizes are rounded to the next 4K multiple of virtual storage. A request for a V=R region is placed on a wait queue until space becomes available if a contiguous space of at least the size of the REGION parameter or the system default is unavailable in virtual or real storage.

Note that V=R regions must be mapped one for one into real storage. Therefore, they do not have their entire virtual storage private area at their disposal; they can use only that portion of their private area having addresses that correspond to the contiguous real storage area assigned to their region, and to LSQA, SWA, and subpools 229 and 230.

Auxiliary Storage Overview

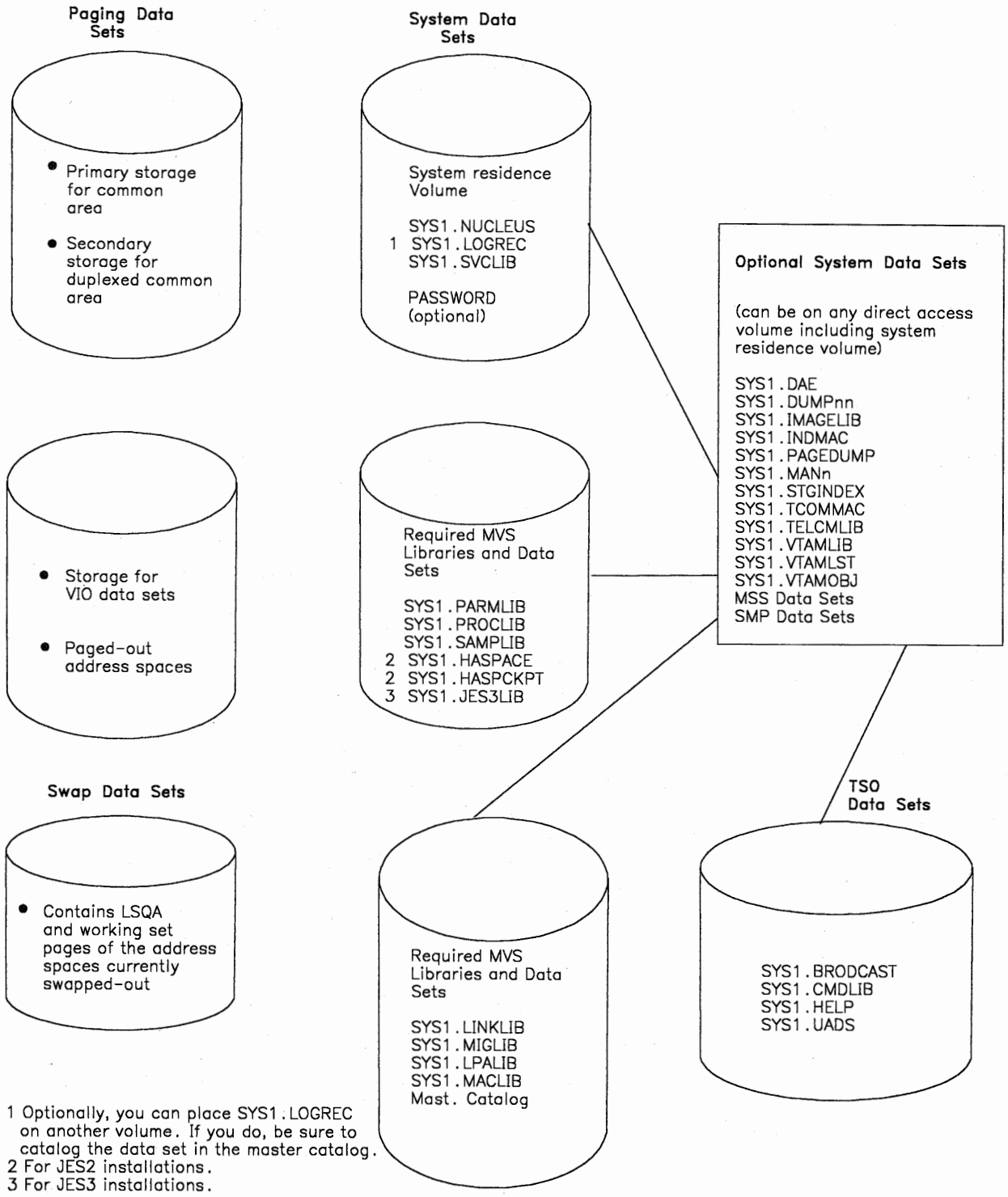
An installation needs auxiliary direct access storage devices (DASD) for housing all system data sets.

Enough auxiliary storage must be available for the programs and data that comprise the system. Auxiliary storage used to support basic system requirements has three logical areas (see Figure 2-3):

- System data set storage area.
- Paging data sets for backup of all pageable address spaces.
- Swap data sets used for LSQA pages and private area pages that are swapped in with the address space (also called "working set").

System Data Sets

Each installation must incorporate required system data sets into the system by allocating space for them on appropriate direct access devices during system generation. The DATASET macro instruction and/or the DEFINE function of access method services is used to define both the storage requirements and the volume for each system data set. Some data sets must be allocated on the system residence volume while some can be placed on other direct access volumes. Special considerations for defining and allocating these data sets are described in *System Generation*.



1 Optionally, you can place SYS1.LOGREC on another volume. If you do, be sure to catalog the data set in the master catalog.
2 For JES2 installations.
3 For JES3 installations.

(**Note** that auxiliary storage need not be organized as shown above. This figure summarizes the makeup of auxiliary storage.)

Figure 2-3. Auxiliary Storage Requirement Overview

Paging Data Sets

Paging data sets contain the paged-out portions of all virtual storage address spaces. In addition, output to virtual I/O devices maybe stored in the paging data sets. Before the first IPL, an installation must allocate sufficient space on paging data sets to back up the following virtual storage areas:

- Primary storage for the pageable portions of the common area
- Secondary storage for duplicate copies of the pageable common area
- The paged-out portions of all swapped-in address spaces - both system and installation
- Space for all address spaces that are, or were, swapped out
- VIO data sets that are backed by auxiliary storage

Initially, paging data sets are specified with the DATASET macro instruction during system generation.

Alternate paging data sets can be specified in IEASYSxx members of SYS1.PARMLIB. When this is done, any PAGE parameter in an IEASYSxx specified during IPL overrides any PAGE parameter in IEASYS00.

To add paging space to the system after system generation, the installation must use the access method services to define and format the new paging data sets. To add the new data sets to the existing paging space, either use the PAGEADD operator command or re-IPL the system, issuing the PAGE parameter at the operator's console. To delete paging space, use the PAGEDEL operator command.

During initialization, paging spaces are set up by merging the selected IEASYSxx parmlib member list of data set names with any names provided by the PAGE parameter (issued at the operator console) and any names from the previous IPL (recorded in records in the PLPA page data set).

Directed VIO

When backed by auxiliary storage, VIO data set pages can be directed to a subset of the local paging data sets through *directed VIO*, which allows the installation to direct VIO activity away from selected local paging data sets and use these data sets only for non-VIO paging. With directed VIO, faster paging devices can be reserved for paging where good response time is important. The NONVIO system parameter, in conjunction with the PAGE parameter in the IEASYSxx parmlib member, enables the installation to define those local paging data sets that are not to be used for VIO, leaving the rest available for VIO activity. However, if space is depleted on the paging data sets that were made available for VIO paging, the non-VIO paging data sets will be used for VIO paging.

The installation uses the DVIO keyword in the IEAOPTxx parmlib member to either activate or deactivate directed VIO. To activate directed VIO, the operator issues a SET OPT=xx command where the xx specifies the IEAOPTxx parmlib member that contains the DVIO=YES keyword; to deactivate directed VIO, xx specifies an IEAOPTxx parmlib member that contains the DVIO=NO keyword. The NONVIO parameter of IEASYSxx is explained more fully in "Part 3: System Initialization." The DVIO keyword of IEAOPTxx is explained in "Part 3: System Initialization" and in "Part 5: The System Resources Manager."

Swap Data Sets

The installation allocates swap data sets to speed up the swapping process. They are used to store the LSQA and working set pages that are currently swapped-out. Swap data sets are not required. However, it is recommended that swap data sets be supplied for maximum system performance. If there is not enough swap data set space or no swap data set space, the LSQA and working set pages normally sent to a swap data set are directed to local page data sets.

The installation specifies its swap data sets with the DATASET macro instruction during system generation, which sets the system SWAP parameter in the IEASYS00 member of SYS1.PARMLIB.

The installation specifies alternate swap data sets in the IEASYSxx member of SYS1.PARMLIB. When this happens, any SWAP parameter in IEASYSxx specified during IPL overrides any SWAP parameter in IEASYS00.

To add swap data set space to the system after system generation, the installation must use the access method services to define and format the new data sets. To add new data sets to the existing swap space, use the PAGEADD operator command with the SWAP parameter. To override those specified in IEASYSxx, re-IPL the system and issue the SWAP parameter, naming the new swap data sets. To delete swap data sets, use the PAGEDEL operator command.

Primary and Secondary PLPA

During initialization, both primary and secondary (or duplexed) PLPAs are established. The PLPA is established initially from the LPALST concatenation.

On the PLPA page data set, ASM maintains records that have pointers to the PLPA and extended PLPA pages. During quick start and warm start IPLs, the system uses the pointers to locate the PLPA and extended PLPA pages. The system then rebuilds the PLPA and extended PLPA page and segment tables, and uses them for the current IPL.

If CLPA is specified at the operator's console, indicating a cold start is to be performed, the PLPA storage is deleted and made available for system paging use. A new PLPA and extended PLPA is then loaded, and pointers to the PLPA and extended PLPA pages are recorded on the PLPA page data set. CLPA also implies CVIO (see below).

The secondary PLPA is saved on the optional duplex paging data set for recovery purposes. If any uncorrectable I/O error occurs on a page-in request from the PLPA, the duplexed PLPA is used.

Virtual I/O Storage Space

Virtual I/O operations may send VIO data set pages to the local paging data set space. During a quick start, the installation uses the CVIO parameter to purge VIO data set pages. During a warm start, the system can recover the VIO data set pages from the paging space. If an installation wants to delete VIO page space reserved during the warm start, it issues the CVIO system parameter at the operator's console. CVIO applies only to the VIO data set pages that are associated with journaled job classes. (The SYS1.STGINDEX data set contains entries for the VIO data sets associated with journaled job classes.) If there are no journaled job classes or no SYS1.STGINDEX data set, there is no recovery of VIO data set pages. Instead, all VIO data set pages are purged and the warm start is effectively a quick start.

If the SPACE parameter for a DD statement having a UNIT parameter, associated with a UNITNAME macro instruction having VIO=YES, is not specified, the default size is 10 primary and 50 secondary blocks with an average block length of 1000 bytes.

No more than 25 paging data sets may be specified during system generation. The cumulative number of page data sets must not exceed 256. This maximum number of 256 page data sets should follow these guidelines:

- There must be exactly one *PLPA* page data set.
- There must be exactly one *common* page data set.
- A *duplex* page data set is optional, but if included, only one can be specified.
- There must be at least one *local* page data set, but no more than 253.

The actual number of pages required in paging data sets depends on the system load, including the size of the VIO data sets being created, the rate at which they are created, the rate at which they are deleted, and how much ESTORE is available and usable for VIO pages.

Part 3. System Initialization

This part of the book contains two sections:

- An overview of initialization
- Parmlib member descriptions

Initialization Overview

System initialization is the part of the system tailoring that takes place after system generation. The tailoring results from the specification of system parameters, first at IPL and then later when certain operator commands are issued.

System Tailoring

System tailoring is the overall process by which an installation selects its operating system. The process consists of the specification of system options through these mechanisms:

- System generation
- MVS configuration program (MVSCP)
- IO configuration program (IOCP)
- Initialization-time selections
- Certain operator commands after IPL
- Implicit system parameters

An installation can identify one or more active versions of the operating system. For instance, the installation might choose to identify a specific version to be used only during off-shift hours.

The operator verifies the version identifier when it appears in the system message IEA101A SPECIFY SYSTEM PARAMETERS. If the installation does not supply a version identifier, the system uses the basic control program (BCP) identifier provided in module IEASYSID.

To identify a version of the operating system, the installation assigns a unique 16-character EBCDIC identifier in IEAVCVT, which resides in the nucleus. The identifier is assigned using the AMASPZAP program immediately after each system generation or creation of an alternate nucleus. For more information on AMASPZAP, see *Service Aids*.

System Generation

System generation involves the specification of particular system options under a driver, before the desired system is available. Some system options are specified by means of parameters in SYSGEN macros, such as DATASET, DATAMGT, and GENTYPE. The SYSGEN macros are documented in *System Generation* .

Some of the DATASET options, for example, establish the initial contents of certain members of SYS1.PARMLIB, such as IEASYS00, and IEAAPP00. Other members of parmlib (IEAABD00, IEADMP00, and LNKLST00) are copied directly from the APARMLIB data set to SYS1.PARMLIB. Although the installation can not determine the initial contents of these members at sysgen, the system programmer

can later alter or enlarge them through the use of the IEBUPDTE utility. (See Figure 3-1 for the complete list of members created by the installation or copied at sysgen.)

MVS Configuration Program

The MVS configuration program (MVSCP) provides a way for an installation to define I/O configurations to MVS. Using the MVSCP, the installation can do the following:

1. Identify the available I/O devices.
2. Define esoteric device groups and name each group.
3. Define eligible device tables (EDTs) and assign estoeric device groups to these tables.
4. Identify devices that are eligible for virtual I/O (VIO).
5. Define changes to the IBM-supplied device preference table.
6. Identify I/O devices that NIP may use as consoles.
7. Request I/O configuration data for the JES3 initialization stream checker.

MVSCP, with other program enhancements, enables the installation to maintain just one copy of the MVS nucleus to support multiple I/O configurations. For more information on MVSCP, see *MVSCP Guide*.

The I/O configuration program (IOCP) allows you to define the I/O configuration to the channel subsystem. You can use a common deck as input to the MVSCP and IOCP processes. For more information, see *IOCP Guide*.

System Tailoring at Initialization Time

Initialization-time choices that help to tailor the system can come from several sources:

- Various types of IPLs.
- Operator entry of parameters from the console.
- SYS1.PARMLIB. This data set is one of the main sources of IPL-time parameters.
- The JES2 initialization data set or JES3 initialization deck.

Types of IPL

There are several types of IPL:

- *Cold Start*: Any IPL that loads (or reloads) the PLPA, but does not preserve VIO data set pages. The first IPL after system generation is always a cold start because the PLPA is initially loaded. Subsequent IPLs are cold starts when the PLPA is reloaded either to alter its contents or to restore its contents if they were destroyed.
- *Quick Start*: Any IPL that does not reload the PLPA and does not preserve VIO data set pages. (NIP resets the page and segment tables to match the last-created PLPA.)
- *Warm Start*: Any IPL that does not reload the PLPA, but does preserve journaled VIO data set pages.

The First IPL After System Generation: At the first IPL after system generation (sysgen), NIP automatically loads the PLPA from the LPALST concatenation. The page data sets for this IPL are those named in parmlib member IEASYS00 (built at sysgen), plus any specified by the operator. These page data sets are those that were specified in SYSGEN DATASET macros that contained the PAGEDSN keyword or the DUPLEXDS,NAME keyword combination.

After the first IPL, the SYS1.LOGREC initialization routine, IFCDIP00, must be run to initialize SYS1.LOGREC. This routine must also be run whenever SYS1.LOGREC is reallocated.

An IPL at Which the PLPA is Reloaded: The PLPA must be reloaded: (1) at the first IPL after system generation, when NIP loads it automatically, (2) at an IPL after the installation has added or modified one or more modules in the LPALST concatenation, has tested the alteration, and now wants to put the replacement module(s) in the PLPA, and (3) at an IPL after the PLPA page data set has been damaged (and is therefore unusable) and its contents must be restored. The PLPA can also be reloaded for other reasons (such as when the addition of more real storage causes the nucleus and the PLPA to overlap). Reloading the PLPA should be discretionary; that is, it should not be a common occurrence. It should be done only when necessary because the associated I/O slows down the IPL and because previously existing VIO data set pages are deleted.

To reload the PLPA from the LPALST concatenation, the operator would enter CLPA (create link pack area) as one of his responses to the SPECIFY SYSTEM PARAMETERS message. (For further information on the loading of the PLPA, see the CLPA parameter in the description of the IEASYSxx parmlib member.)

An IPL After Power-Up: The IPL performed after power-up is called a "quick start," because the PLPA from the previous IPL can be used without reloading from the LPALST concatenation. For a "quick start," the CVIO system parameter is used; VIO data set pages are purged, page data sets added (optionally reserved for non-VIO paging), and swap data sets replaced. The operator or the IEASYSxx parmlib member can add additional page data sets by specifying the PAGE parameter (with or without the NONVIO system parameter) and can replace swap data sets by specifying the SWAP parameter. (For information on the CVIO, PAGE, and NONVIO parameters, see the description of the IEASYSxx parmlib member later in this chapter.)

An IPL After a System Crash: Provided that the operator does not enter the CLPA or CVIO system parameters, the operator can "warm start" the system after a system crash. Existing journaled VIO data set pages and PLPA pages are retained by the auxiliary storage manager for continued use. The specified parmlib parameter list (IEASYSxx) would not include the CVIO or CLPA system parameters. (The specification of one or more IEASYSxx members by the operator at IPL time is described in the next topic, "Operator Entry of Parameters.")

Any definitions of existing page data sets as non-VIO local page data sets are preserved. Also the operator can define a local page data set that previously was used for VIO paging as a non-VIO local page data set. During system operation, the VIO pages on the newly designated non-VIO local page data set will migrate to any local page data set used for VIO paging. An installation can remove a local page data set that was designated on the previous IPL as a non-VIO local paging data set. Removing a local page data set prior to a "warm start" requires that the local page data set contain no VIO pages. If the local page data set contains VIO

pages then the "warm start" is changed into a "quick start."[1] For more information on designating non-VIO page data sets see the NONVIO system parameter in the IEASYSxx parmlib member description later in this part of the manual.

Operator Entry of Parameters

The operator responds to the SPECIFY SYSTEM PARAMETERS message to direct NIP, master scheduler initialization, and other components to the desired parmlib members. The operator can select the default general parameter list IEASYS00, or enter SYSP=(aa,bb...) to select one or more alternate general parameter lists, such as IEASYS01, IEASYS02, etc. The alternate lists can supplement or partially override the basic list. The operator need not enter parameter values directly, except for those cases in which parameters are missing, are syntactically invalid, can't be read, or must be supplemented to satisfy a special case. (An example of a special case would be the operator entry of the PAGE parameter to increase the amount of paging space.)

If an error occurs with certain parmlib members, the operator is prompted to manually enter one or more of the member's parameters. If the parameter can't be corrected, the operator can accept the system defaults. Most parameters have defaults, either as default parmlib members, or as coded values in system components. If a default doesn't exist (and if a parameter is not required), the operator can cancel the parameter. (The defaults are listed in the individual descriptions of parmlib members later in this chapter.)

Note:

To request a continuation line when manually entering a parameter, use a comma c (,c) or a comma blank (,) at the end of the response line.

An operator-entered parameter overrides the same parameter specified in parmlib member IEASYS00 or IEASYSxx, except for:

- A parameter for which operator intervention is prohibited (OPI=NO). In this case, the operator-entered parameter is ignored (unless the parmlib parameter was syntactically invalid and is being corrected from the console).
- The PAGE parameter. The page data set names entered by the operator are added for the life of the IPL to those specified in either IEASYS00 or IEASYSxx. (For information on the PAGE parameter, refer to the description of member IEASYSxx later in this chapter.)

---

[1] A local page data set specified as NONVIO can contain VIO pages if one of the following conditions exist:

- Directed VIO was turned off while the previous system was active.
- A warning message to the operator indicated that VIO pages spilled to a non-VIO data set because no more space was available on those page data sets that were being used for VIO.

The Use of SYS1.PARMLIB

SYS1.PARMLIB is read by NIP and master scheduler initialization at IPL, and later by such components as the system resource manager, the TIOC, and GTF, which are invoked by operator commands.[2] The purpose of parmlib is to provide many initialization parameters in a pre-specified form in a single data set, and thus minimize the need for the operator to enter parameters. The SYS1.PARMLIB data set can be blocked and can have multiple extents, but it must reside on a single volume.

Parmlib contains both a basic or default general parameter list IEASYS00 and possible alternate general parameter lists, called IEASYSaa, IEASYSbb, etc. Parmlib also contains specialized members, such as COMMNDxx, and IEALPAxx. Any general parameter list can contain both parameter values and "directors." The directors (such as MLPA=01) point or direct NIP or master scheduler initialization to one or more specialized members, such as IEALPA01.

Member IEASYS00, the default general parameter list, is always read, but its contents can be overridden and/or augmented by one or more alternate general parameter list. IEASYS00 can be further supplemented and/or partially overridden by operator-entered parameters. The IEASYSxx lists are selected by the operator through the SYSP parameter at IPL. The specialized members can be named by the general parameter lists (IEASYS00 and IEASYSxx), or named by the operator at IPL.

If the same parameter appears in both IEASYS00 and a specified alternate IEASYSxx list, the value in the alternate list overrides. In addition, a parameter value in a later specified IEASYSxx list overrides the same parameter in an earlier specified list. For example, assume that the operator enters R 00,SYSP=(01,02) in order to select the two parameter lists IEASYS01 and IEASYS02. Further assume that these two lists and IEASYS00 contain these values:

IEASYS00: ...,MLPA=00,...

IEASYS01: ...,MLPA=(01,02),...

IEASYS02: ...,MLPA=03,SQA=10,...

From the above values, NIP accepts: MLPA=03,SQA=10.

Note: The CLPA, CVIO, OPI, and PAGE parameters are exceptions; for more information see their descriptions later in this section.

If a particular parameter or member is unavailable or incorrect, the system, depending on the particular member:

- Uses a default value.
- Bypasses the parameter or the rest of the member.
- Prompts the operator to enter replacements for the invalid parameter(s), or to enter all the parameters in the member, or to re-IPL, or to cancel the parameter or member by entering ENTER.

---

[2] The TIOC is the terminal I/O coordinator, whose parameters are described under member IKJPRM00. GTF is the generalized trace facility, whose parameters are described under member GTFPARM.

The handling of each member if a syntax error or a read error occurs is listed later in this overview in Figure 3-2.

How Parmlib Members Are Created: Parmlib members are created in several ways:

- Some are unconditionally created at SYSGEN by being copied from the APARMLIB data set. They may later be changed or augmented by the installation through the use of the IEBUPDTE utility.
- A few are conditionally created at sysgen, if particular SYSGEN macros and keywords are specified.
- The remaining members can be explicitly created by the installation.

Figure 3-1 shows which parmlib members are created at sysgen, whether the creations are conditional, the names of any associated SYSGEN macros and keywords, and the IPL-time parameters that direct the reading system components to the desired specialized members. (See the notes at the bottom of figure.)

Overview of Parmlib Members: Figure 3-2 lists valid parmlib members. The table briefly describes the purpose of each member and lists additional categories of information for each.

Figure 3-1 (Page 1 of 3). Parmlib Members: Relationships to IPL Parameters and SYSGEN Parameters

Member	Associated IPL-Time Parameter (in IEASYS00, IEASYSxx, or entered by the Operator)	Initially Built by Installation	SYSGEN Macro and Parameter
ADYSET00	none	no: default list is copied from APARMLIB	Not applicable
ADYSETxx	none	yes	Not applicable
BLSCECT	BLSCECT	no: default list is copied from APARMLIB	Not applicable
CLOCK00	CLOCK=00	no: default list is copied from APARMLIB	Not applicable
CLOCKxx	CLOCK=xx	no: default list is copied from APARMLIB	Not applicable
COFVLFxx	none	no	Not applicable
COMMNDxx	CMD=xx	yes	Not applicable
CONFIGxx	none	yes	Not applicable
CONSOLxx	CON=xx	yes	Not applicable
CSVLLAxx	none	yes	Not applicable
EXSPATxx	none	yes	Not applicable
GRSCNFxx	GRSCNF=xx	yes	Not applicable

Figure 3-1 (Page 2 of 3). Parmlib Members: Relationships to IPL Parameters and SYSGEN Parameters

Member	Associated IPL-Time Parameter (in IEASYS00, IEASYSxx, or entered by the Operator)	Initially Built by Installation	SYSGEN Macro and Parameter
GRSRNL00	GRSRNL=00	no: default list is copied from APARMLIB	Not applicable
GRSRNLxx	GRSRNL=xx	yes	Not applicable
GTFPARM	none	no: default list is copied from APARMLIB.	Not applicable
IEAABD00	none	no: default list is copied from APARMLIB.	Not applicable
IEAAPF00	APF=00	yes	Not applicable
IEAAPFxx	APF=xx	yes	Not applicable
IEAAPP00	none	yes	Not applicable
IEACMD00	none	no: default list is copied from APARMLIB.	Not applicable
IEADMP00	none	no: default list is copied from APARMLIB.	Not applicable
IEADMR00	none	no: default list is copied from APARMLIB.	Not applicable
IEAFIX01	FIX=01	not if sysgen macro is specified.	DATASET RESIDNT=
IEAFIXxx	FIX=xx	yes	Not applicable
IEAICS00	ICS=00	yes	Not applicable
IEAICSxx	ICS=xx	yes	Not applicable
IEAIPS00	IPS=00	no: default list is copied from APARMLIB.	Not applicable
IEAIPSxx	IPS=xx	yes	Not applicable
IEALPAxx	MLPA=xx	yes	Not applicable
IEAOPT00	OPT=00	no: default list is copied from APARMLIB.	Not applicable
IEAOPTxx	OPT=xx	yes	Not applicable
IEAPAK00	PAK=00, although member is used only when CLPA is specified	yes: optionally supplied by installation before IPL.	Not applicable
IEAPAKxx	PAK=xx	yes	Not applicable

Figure 3-1 (Page 3 of 3). Parmlib Members: Relationships to IPL Parameters and SYSGEN Parameters

Member	Associated IPL-Time Parameter (in IEASYS00, IEASYSxx, or entered by the Operator)	Initially Built by Installation	SYSGEN Macro and Parameter
IEASLPxx	SLP=xx	yes	Not applicable
IEASVCxx	SVC=xx	yes	Not applicable
IEASYS00	none	no, if SYSGEN macros are specified.	(see Figure 3-6)
IEASYSxx	SYSP=xx, issued by the operator	yes	Not applicable
IECIOSxx	IOS=xx	no	Not applicable
IEFSSNxx	SSN=xx	no	Not applicable
IGDSMSxx	ID=xx in IEFSSNxx	yes	Not applicable
IKJPRM00	none	no	Not applicable
IKJTSO00	none	yes	Not applicable
IPCSPRxx	IPCS=xx, issued by the operator	yes	Not applicable
LNKLST00	LNK=00	no: default list is copied from APARMLIB.	Not applicable
LNKLSTxx	LNK=xx	yes	Not applicable
LPALSTxx	LPA=xx	yes	Not applicable
MPFLSTxx	none	yes	Not applicable
MVIKEY00	none	no: default list is copied from APARMLIB	Not applicable
PFKTABxx	PFK=xx	yes	Not applicable
SCHEDxx	SCH=xx	yes	Not applicable
SMFPRM00	SMF=00	no: default list is copied from APARMLIB	Not applicable
SMFPRMxx	SMF=xx	yes	Not applicable
TSOKEY00	none	yes	Not applicable
VATLST=xx	VAL=xx	yes	Not applicable

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
ADYSET00(xx)	Parameters that control dump analysis and elimination (DAE) processing.							
	yes	optional	yes	Both IPL and SET DAE command.	no	Error message requires the operator to correct the parmlib member SET DAE command.	Error message requires the operator to start DAE again (using the SET DAE command) after the problem is fixed.	N/A
BLSCECT	Parameters that control dump formatting exit routines. Used by IPCS and AMDPRDMP to build the exit control table (ECT).							
	yes	optional	no	command	no			N/A
CLOCK00 (xx)	Parameters prompt the operator to set the TOD clock during NIP and specifies the difference between the local time and GMT.							
	yes	optional	no	IPL	no			N/A
COFVLF00(xx)	Allows an authorized program to store named objects in virtual storage managed by VLF.							
	yes	required	yes	Both IPL and START VLF command.	no	Error message is issued.	Error message is issued.	Error message is issued.
COMMND00(xx)	Commands to be issued by the control program immediately after initialization. Also contains a parameter that controls prompting during TOD clock initialization. JES commands may not be included.							
	no	optional	no	IPL	no	Ignores invalid command name. TOD defaults to NOPROMPT.	No commands are processed.	N/A

Figure 3-2 (Part 1 of 9). Characteristics of Parmlib Members

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
CONFIG00(xx) Allows the installation to define a standard configuration that is compared with current configuration and to reconfigure processors, storage, and channel paths.								
	no	optional	no	DISPLAY M command with CONFIG option and CONFIG command with MEMBER option.	no	Invalid commands are ignored and an error message is issued.	Processing is terminated and an error message is issued.	N/A
CONSOL00(xx) Parameters to define an installation's console configuration, theinitialization values for communications tasks, the default routing codes for all WTO/WTOR messages that have none assignedpnd the characteristics for the hardcopy log.								
	no	required	no	IPL	Yes, if L option is specified with the CON parameter in IEASYSxx or by the operator.	Error message is issued.	The operator is prompted for a new CONSOLxx member.	Error message.
CSVLLA00(xx) Allows an installation to list the entry point name or LNKLST libraries that can be refreshed by the 'MODIFY LLA, UPDATE=xx' command.								
	no	optional	no	MODIFY LLA command	yes	Error message is issued.	Error message is issued.	Error message is issued.
EXSPAT00(xx) Allows an installation to specify actions to be taken to recover from excessive spin conditions without operator involvement.								
	no	optional	no	Both IPL and SET EXS command	no	Error message is issued.	Error message is issued.	Error message is issued.

Figure 3-2 (Part 2 of 9). Characteristics of Parmlib Members

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
GRSCNF00(xx) Parameters that identify what systems are to share resources in a global resource serialization complex.								
	no	. required only if GRS=START or GRS=JOIN is specified. · ignored if GRS=NONE is specified.	yes, by specification of the RESMIL parameter	IPL	no	Error message requires the operator to either correct the parmlib member and restart the system or reply NONE.	Error message requires the operator to either restart the system after the problem is fixed or reply NONE.	Same as with syntax error.
GRSRNL00(xx) Resource name lists (RNLs) that the system uses when a global resource serialization complex is active.								
	yes	. required only if GRS=START or GRS=JOIN is specified. · ignored if GRS=NONE is specified.	no	IPL	no	Error message requires the operator to either correct the parmlib member and restart the system or replay NONE.	Error message requires the operator to either restart the system after the problem is fixed or reply NONE.	N/A
GTFPARM Parameters to control GTF is started.								
	yes	Optionallly used. Built automatically at sysgen. Can be modified by installation.	Indirectly because SRM sysevent trace option is a tuning aid.	START GTF	Yes, automatically at START GTF.	On any error, prompts for all parameters.	Same as with syntax error.	N/A
IEAABD00 Default parameters for an ABEND dump when a SYSABEND DD statement has been specified.								
	yes	Optional. However, if member is unavailable, ABEND dumps may not be possible without DUMPOPT lists.	no	IPL	no	Message lists valid parameters that were accepted. Invalid parameters are rejected.	Error message. Parameters are rejected.	N/A
IEAAPF00(xx) Names of authorized program libraries.								
	no	optional (SYS1.LINKLIB and SYS1.SVCLIB are always authorized.)	no	IPL	no	Prompts operator to respecify bad parameters or cancel parameters via ENTER key.	Same as with syntax error.	N/A
IEAAPP00 Names of authorized installation-written I/O appendage routines.								
	no	optional	no	IPL	no	Error message. Partial appendage name table is built, if possible.	Same as with syntax error.	N/A

Figure 3-2 (Part 3 of 9). Characteristics of Parmlib Members

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
IEACMD00	IBM-supplied commands that are processed during system initialization.							
	yes	optional	yes	IPL	no	Ignores invalid command name.	No commands are processed.	N/A
IEADMP00	Default parameters for an ABEND dump when SYSUDUMP DD statement has been specified.							
	yes	Optional. However, if member is unavailable, ABEND dumps may not be possible without DUMPOPT lists. Built automatically at sysgen. Can be modified by installation.	no	IPL	no	Message lists valid parameters that were accepted invalid parameters are rejected.	Error message. Parameters are rejected.	N/A
IEADMR00	Default parameters for an ABEND dump when a SYSMDUMP DD statement has been specified.							
	yes	Optional. However, if member is unavailable ABEND dumps may not be possible without DUMPOPT lists. Built automatically at sysgen. Can be modified by installation.	no	IPL	no	Message lists valid parameters that were accepted. Invalid parameters are rejected.	Error message. Parameters are rejected.	N/A
IEAFIX00(xx)	Names of modules from SYS1.SVCLIB, the LNKLST concatenation, and the LPALST concatenation to be fixed in real storage for the duration of the IPL.							
	IEAFIX00-yes IEAFIXxx-no	IEAFIX00 is created optionally at sysgen.	yes	IPL	yes, if L option is specified with FIX parameter in IEASYS00(xx), or by the operator.	Prompts operator to respecify bad parameters or cancel then via ENTER key.	Same as with syntax error.	Error message. Obsolete module names are ignored.

Figure 3-2 (Part 4 of 9). Characteristics of Parmlib Members

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
IEAICS00(xx)	Parameters of an installation control specification that associate units of work (transactions) with performance groups. Performance groups are used by the system resources manager to control and report on transactions.							
	no	optional	yes	both IPL and SET ICS command.	yes, if L option is specified with the ICS = xx parameter in IEASYS00(xx), or by the operator.	Prompts operator to respecify or cancel parameter via ENTER key. If the parameter is cancelled, no IEAICSxx member is put into effect; performance groups are assigned by means of the JCL or LOGON PERFORM parameter value.	Same as with syntax error.	N/A
IEAIPS00(xx)	Parameters of an installation performance specification that control workload manager of system resources manager.							
	IEAIPS00-yes IEAIPSxx-no	IEAIPS00=req'd. IEAIPSxx=opt.	yes	both IPL and SET IPS command.	yes, if L option is specified with IPS parameter in IEASYS00(xx), or by the operator.	Prompts operator to respecify alternate IPS member or to default to IEAIPS00. If no IEAIPS00 or IEAIPS00 is invalid uses system defaults.	Same as with syntax error.	N/A

Figure 3-2 (Part 5 of 9). Characteristics of Parmlib Members

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
IEALPA00(xx)	Names of reenterable modules from SYS1.SVCLIB, the LNKLST concatentation, and the LPALST concatenation that are to be loaded as a temporary extension to the PLPA.							
	no	optional	yes	IPL	yes, if L option is specified with the MLPA parameter in IEASYS00(xx), or by the operator.	Prompts operator to respecify, or cancel parameter via ENTER key.	Same as with syntax error.	Error message. Ignores obsolete module name.
IEAOPT00(xx)	Parameters that control resource and workload management algorithms in the system resources manager.							
	IEAOPT00-yes IEAOPTxx-no	optional	yes	both IPL and SET OPT command.	yes, if L option is specified with the OPT parameter in IEASYS00(xx), or by the operator.	Prompts operator to respecify or cancel parameter via ENTER key. If parameter is cancelled, default values are used.	Same as with syntax error.	N/A
IEAPAK00(xx)	"Pack List" names of groups of modules in the LPALST concatenation that NIP will load between page boundaries to minimize page faults.							
	no. Optionally, installation can provide before IPL.	optional	yes	IPL	no	Bypasses the pack group that contains the error. Processes the next pack group.	The pack groups read before the errors are processed. Other pack groups are omitted.	Error message. Ignores obsolete module names.
IEASLPxx	Contains valid SLIP commands.							
	no	optional	no	SET SLIP command	no	Normal SLIP response to prompt the operator.	Processing stops.	Syntax error message is issued.
IEASVCxx	Allows the installation to define its own SVCs in the SVCTABLE.							
	no	optional	no	IPL	Yes, if L option is specified with the SVC porameter in IEASYS00(xx), or by the operator.	Message is issued and the statement in error is not processed. Processing continues with the next statement.	Same as with syntax error.	Same as with syntax error.

Figure 3-2 (Part 6 of 9). Characteristics of Parmlib Members

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
IEASYS00(xx)	System parameters that are valid responses to the SPECIFY SYSTEM PARAMETERS message. Multiple system parameter lists are valid. The list is chosen by operator SYSP parameter. IEASYS00, the default member, is initially built at sysgen, although modifiable later by the installation.							
	yes. Built at sysgen from macros.	See Note 2.	See Note 3.	IPL	See Note 1. (L parameter must be specified with parameter or member.)	Prompts operator to respecify, or cancel parameter via ENTER key. Parameters processed before the error are retained.	Parameters processed before error are retained. Operator is asked to specify an alternate member. If he does, new parameters override those retained.	Obsolete parameters are treated as syntax errors.
Notes:								
1. These parameters can be listed at IPL: APF, DUMP, FIX, ICS, IPS, MLPA, SYSP, and OPT.								
2. The only mandatory parameter is PAGE. Other parameters have coded defaults.								
3. Performance-oriented parameters in IEASYS00(xx): APG, CMD, FIX, IPS, MLPA, OPT, REAL, RSU, WTOBFRS, WTORPLY.								
IECIOS00(xx)	Parameters that control missing interrupt handler (MIH) time intervals and update hot I/O detection table (HIDT) values.							
	no	optional	no	IPL	no	Message is issued and default value is substituted.	Same as with syntax error.	Obsolete parameters are treated as syntax errors.
IEFSSN00(xx)	Parameters that identify what subsystems are to be initialized.							
	yes	required	yes, depending on the function of each subsystem.	IPL	no	Message issued identifying erroneous record. Next record is read and processed.	Processing of the parmlib members is terminated and an error message is issued.	N/A
IKJPRM00	TIOC parameters that are used to control TSO/TCAM time sharing buffers.							
	no	optional	yes	MODIFY tcamproc command, if TS=START is specified in command.	no	Default value is substituted.	Same as with syntax error.	Obsolete parameters are ignored.
IKJTSO00	For TSO/E Release 4 specifies authorized commands and authorized programs, programs that are authorized when called through the TSO service facility, commands that may not be issued in the background, and defaults for SEND and LISTBC processing.							
	yes, in SYS1.SAMPLIB	optional	no	IPL	no	Error message is issued. Processing continues.	Same as with syntax error.	N/A
IPCSPR00(xx)	Parameters that are used during an IPCS session.							
	yes	optional	no	IPCS command initialization	no	Error message is issued and IPCS sessions continues.	Same as with syntax error.	Same as with syntax error.

Figure 3-2 (Part 7 of 9). Characteristics of Parmlib Members

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
LNKLST00(xx)	List of data sets to be concatenated to SYS1.LINKLIB to form the LNKLST concatenation.							
	yes (Contains only SYS1.LINKLIB)	optional	no	IPL	Yes, if L option is specified with LNK parameter in IEASYS00(xx), or by the operator.	Prompts operator to respecify, or cancel parameter via ENTER key.	No data sets are concatenated to SYS1.LINKLIB. Operator can re-IPL to specify an alternate LNKLSTxx member.	Error message. Data sets not found will not be concatenated to SYS1.LINKLIB.
LPALSTxx	List of data sets to be concatenated to SYS1.LPALIB from which the system builds the pageable LPA (PLPA).							
	no	optional	no	IPL	Yes, if L option is specified with LPA parameter in IEASYSxx or by the operator.	Prompts operator to respecify, or cancel parameter using ENTER key.	No data sets are concatenated to SYS1.LPALIB. Operator can re-IPL and specify an alternate LPALSTxx member.	N/A
MPFLST00(xx)	Parameters that the message processing facility (MPF) uses to control message processing and display.							
	no	optional	no	SET MPF command	Yes, via the DISPLAY command.	Message issued for each invalid member.	Parameters are rejected.	Sames as with syntax error.
MVIKEY00	Parameters to control MSSC data and messages.							
	yes Automatically copied at sysgen. Can be modified by installation.	required for MSS	yes	IPL	Yes, at IPL. If error occurs, default is taken.	Prompts operator to enter parameter or re-IPL.	Same as with syntax error.	Same as with syntax error.

Figure 3-2 (Part 8 of 9). Characteristics of Parmlib Members

Member	Supplied by IBM	Required or Optional	Directly affects performance	Read at IPL or at command	Allows listing of parameters at IPL or command	Response to Errors (N/A = not applicable)		
						Syntax Error	Read Error	Unsupported Parameters
PFKTABxx	Parameters contain the definitions for program function key tables (PFK tables).							
	no	optional	no	IPL or SET PFK command.	no			N/A
SCHEDxx	Provides centralized control over which EDT the system is to use, the size of the master trace table, the completion codes to be eligible for automatic restart and programs to be included in the PPT.							
	no	optional	no	IPL	Yes, if (L) is specified in IEASYSxx or in response to the specify system parameters prompt.	Diagnostic error message is issued. Different processing is done for each statement type in IEASYSxx.	Diagnostic error message is issued and the next record is read.	
SMFPRM00(xx)	Parameters that define SMF options.							
	PRM00-yes PRMxx-no	PRM00=req'd. PRMxx=opt.	yes	IPL or SET SMF command.	Yes, if PROMPT (list) or PROMPT (ALL) parameter is specified.	Prompts operator to enter parameter or re-IPL.	Same as with syntax error.	Obsolete parameters are ignored.
TSOKEY00	VTIOC parameters that are used by TSO/VTAM time sharing.							
	no	optional	yes	START TSO command.	Yes, automatically at START TSO.	Deafult value is substituted.	Deafult value is substituted.	N/A
VATLST00(xx)	Volume attribute list that defines the "mount" and "use" attributes of direct access volumes.							
	no	optional	yes	IPL	No. Operator has option to get list only if error occurs.	Error message. Bypasses bad entry. Processes remaining entries.	Operator is given choice of processing remaining list(s) if multiple lists, or specifying new VATLST VATLSTxx member, or re-IPLing.	N/A

Figure 3-2 (Part 9 of 9). Characteristics of Parmlib Members

How to Control Parmlib: To control parmlib and assure that it is manageable, you should consider the following problem areas and suggested solutions:

- Delete unsupported parameters and members. Because most components treat unsupported parameters from previous releases as syntax errors, you should probably remove the old parameters or build parmlib from scratch. This action will minimize the need for operator responses during an IPL. Furthermore, you can save space by removing unsupported members.
- Use the parmlib members for the appropriate functions. For example, use COMMNDxx to contain commands useful at system initialization. Use IEACMDxx for IBM-supplied commands. Use IEASLPxx for SLIP commands. See each member for further information.
- Update parmlib with new and replacement members, as you gain familiarity with the new release. You can use the IEBUPDTE utility to add or replace members. Figure 3-3 illustrates the JCL for adding two new members (IEASYS05 and IEASYS06) and replacing one old member (IEAPAK04). (Consult the *Utilities* manual for further information on the use of IEBUPDTE.) To prevent excessive growth of parmlib, use the "compress" function of IEBCOPY to delete obsolete data.

```
//ADDLISTS          JOB     61938,'R. L. WILSON'
//STEP              EXEC    PGM=IEBUPDTE,PARM=MOD
//SYSPRINT          DD      SYSOUT=A
//SYSUT1            DD      DSNAME=SYS1.PARMLIB,DISP=OLD
//SYSUT2            DD      DSNAME=SYS1.PARMLIB,DISP=OLD
//SYSIN             DD      DATA
./                  REPL NAME=IEAPAK04,LEVEL=02,SOURCE=1,LIST=ALL
./                  NUMBER NEW1=01,INCR=02
                    (IEFAB400,IGG0325A,IGG0325H,IGC0003B),
                    (IGG0325B,IGG0325D,IGG0325E),
                    IGG0235G),(IFG0202J,IFG0202K,IFG0202L)
./                  ADD NAME=IEASYS05,LIST=ALL
./                  NUMBER NEW1=01,INCR=02
                    MLPA=(00,01),SQA=2
./                  ADD NAME=IEASYS06,LIST=ALL
./                  NUMBER NEW=01,INCR=02
                    MLPA=(02,03),
                    SQA=1,FIX=(00,01),OPI=NO
./                  ENDUP
/*
```

Note: This example shows the format of IEBUPDTE statements, not the content of parmlib members.

Figure 3-3. Example of Adding and Replacing Parmlib Members by Means of the IEBUPDTE Utility

- Keep track of which parameters were specified at SYSGEN (through DATASET macro) and which parameters are included in particular parmlib members. This bookkeeping is necessary for two reasons: 1. The system doesn't keep track of parmlib members and their parameters. 2. The default general parameter list IEASYS00 is always read by NIP and master scheduler initialization. The parameters in IEASYS00 can be overridden by the same parameters when they are specified in alternate general lists, such as IEASYS01, or IEASYS02. Furthermore, certain parameters, such as FIX, APF, and MLPA, direct the system to particular specialized members (in this example, IEAFIXxx, and IEALPAxx). The installation should keep records of which parameters and which values are in particular members, and which general members point to which particular specialized members (COMMNDxx, IEALPAxx, etc.) A grid or matrix for such bookkeeping is very helpful.
- Allocate sufficient space for parmlib. One way to estimate space is to count the number of 80-character records in all members and factor in the blocksize of the data set. Then add a suitable growth factor (e.g., 100-300%) to allow for future growth of alternate members. Consult Figure 3-2 to determine which members can have multiple alternates. To recapture space occupied by deleted members, use the "compress" function of IEBCOPY.
- Decide the volume and device that should hold parmlib. The volume could be demountable, although it must be mounted in order for the operator to IPL, to start GTF, to start RMF, to start TSO by use of the MODIFY tcamproc command, or to specify that a new parmlib member is to take effect by the use of the SET command. The data set must be cataloged, unless it resides on SYSRES. It could be placed on a slow or moderate speed device.
- Password protect the data set. Parmlib should be password protected for "write." The purpose is to preserve system integrity by protecting the appendage member (IEAAPP00) and the authorized program facility member (IEAAPFxx) from user tampering.

General Syntax Rules for the Creation of Members: The following general syntax rules apply to the creation of most parmlib members. Exceptions to these rules are described under specific members later in this chapter. The general rules are:

- Logical record size is 80 bytes.
- Blocksize must be a multiple of 80.
- Any columns between 1 and 71 may contain data.
- Columns 72 through 80 are ignored.
- Continuation is indicated by a comma followed by one or more blanks after the last entry on a record.
- Leading blanks are suppressed. A record therefore need not start at a particular column.
- Suffix member identifiers (such as LNK = A2) can be any alphameric combination.

Causing an Alternate Nucleus Substitution

Another less common way to change the system at an IPL is to cause the IPL program to read a member of a nucleus data set that is different from IEANUC01, the default nucleus member. One reason for such a nucleus switch may be the need to apply a PTF to the nucleus. You can IPL a secondary (alternate) nucleus by either of the following methods:

- Editing the alternate nucleus character of the load parameter string before selecting the "initialize SCP" function. Use the SYSCTL (CC012) frame of the system console. Modify the first character of the eight-character parameter string to specify the suffix for IEANUC0x. The IPL program retrieves this character from the system console frame and concatenates it as a suffix to IEANUC0 to form the alternate SYS1.NUCLEUS member name.
- Stopping the IPL function in time to alter the default contents of location X'08', which is X'F1', to the desired alternate SYS1.NUCLEUS name, then continuing the IPL function. IPL uses the designated alternate nucleus character at location X'08' and concatenates it as a suffix to IEANUC0 to form the alternate SYS1.NUCLEUS member name. The IPL function can be stopped by placing an address-compare-stop on the first executed instruction of IEAIPL00 (location X'154') or by placing the processor in instruction step mode.

Specifying an Alternate Master (System) Catalog

Another way to change the system at an IPL is to cause NIP to read a member of SYS1.NUCLEUS that is different from SYSCATLG, the default member. An alternate master catalog can be selected by:

- Responding to system message IEA347A SPECIFY MASTER CATALOG PARAMETER with a two-character reply. The two characters are appended to SYSCAT to form the member name to be read by NIP.

For additional information, see *System Commands*.

System Tailoring Through Operator Commands

After IPL, several operator commands provide additional system tailoring by directing particular groups of parameters to specific system components. The commands include:

- START tcamproc
- MODIFY tcamproc
- START TSO
- SET DAE
- SET ICS
- SET IPS
- SET OPT
- SETDMN
- SET MPF
- SETSMF
- SET SMF
- SET SMS
- SETSMS
- SET SLIP
- SET PFK
- START GTF
- START vtamproc

- START VLF,SUB=MSTR,NN=xx
- START LLA
- MODIFY LLA
- STOP LLA
- MODE command

For additional information, see *System Commands.*

Starting TCAM

The operator can enter TCAM initialization parameters when he starts TCAM, as a response to the SPECIFY TCAM PARAMETERS message. TCAM issues the message if the system programmer accidentally or deliberately omitted one or more required parameters when he coded the INTRO macro for TCAM assembly. In response to the message, the operator can add to or modify existing TCAM parameters. (For information on the TCAM parameters, see *TCAM Installation.*)

Starting VTAM

VTAM start options can be entered by the network operator as parameters in the START command or they can be specified in a start option list. The start option list is stored in SYS1.VTAMLST and is specified by the LIST parameter in the START command. For information about VTAM initialization, see *VTAM Planning and Installation.*

Starting TSO/TCAM Time Sharing

The operator starts TCAM, then issues the following command to start time sharing:

```
MODIFY tcamproc, TS=START [,member name]
```

The optional parameter member name can specify an installation-defined parmlib member, or can default to the parmlib member IKJPRM00. In either case, the member is read by the terminal I/O coordinator (TIOC). The TIOC uses the parameters mainly to control time-sharing buffers. (For additional information on TIOC parameters, see the description of member IKJPRM00 later in this chapter.)

Starting TSO/VTAM Time Sharing

The operator starts VTAM, then issues the START command to start TSO/VTAM time sharing. The optional parameter MEMBER can be used to specify an installation-defined parmlib member, or the default parmlib member TSOKEY00 can be used. The member is read by the terminal control address space (TCAS) and the values are placed into the TCAS table. (If the member cannot be read, default values in the TCAS program are used.) The VTAM terminal I/O coordinator (VTIOC) uses these values mainly to control time sharing buffers. (For additional information on TSO/VTAM time sharing parameters, see the description of member TSOKEY00 later in this chapter.)

Using SET DAE to Change Dump Analysis and Elimination Parameters

The SET DAE command allows the installation to specify the ADYSETxx parmlib member to be used by dump analysis and elimination (DAE). With this command, the installation can dynamically change the DAE parameters without reinitializing the system.

For more information, see the description of the ADYSETxx parmlib member later in this chapter.

Using SET ICS to Change System Resources Manager Parameters

The SET ICS command causes the system resources manager to receive a specific installation control specification (parmlib member IEAICSxx). Using this command, the installation can dynamically change the installation control specification parameters without reinitializing the system. (For more information, see the ICS parameter in the description of the IEASYSxx member, and in "Part 5: The System Resources Manager.")

Using SET IPS to Change System Resources Manager Parameters

The SET IPS command causes the system resources manager to receive a specific installation performance specification (IPS). This IPS is a parmlib member (IEAIPSxx). Using this command, the installation can dynamically change IPS parameters without reinitializing the system. (For more information, see the IPS parameter in the description of the IEASYSxx member, and in "Part 5: The System Resources Manager.")

Using SET OPT to Change System Resources Manager Parameters

The SET OPT command causes the system resources manager to receive a specific IEAOPTxx parmlib member. Using this command, the installation can dynamically change OPT parameters without reinitializing the system. (For more information, see the OPT parameter in the description of the IEASYSxx member, and in "Part 5: The System Resources Manager.")

Using SETDMN to Change Domain Constraints

The SETDMN command provides a way for altering the constraint values on a domain's multiprogramming level. The information from this command is valid only for the life of the IPL - it does not change fields in the IPS member.

Using SET MPF to Control Message Processing and Display

The SET MPF command selects an MPFLSTxx parmlib member that contains information that the message processing facility (MPF) uses to control message processing and message display.

This command provides a way to suppress nonessential message traffic to the operator's console, causing only selected messages to be displayed. In response to operating conditions, the SET MPF command can be used repeatedly to change the amount of nonessential message traffic being suppressed. In this way, the operator can tailor the message traffic to suit the operational needs of the installation. SET MPF can also be issued through the COMMNDxx parmlib member during system initialization.

Using SET SMF to Change SMF Parameters or Restart SMF

The SET SMF command allows the installation to specify the SMFPRMxx parmlib member to be used by the system management facilities (SMF) or to restart SMF if it terminates. Using this command, the installation can dynamically change to another SMFPRMxx member or can restart SMF without reinitializing the system.

Using SETSMF to Change SMF Parameters

In contrast to the SET SMF command, which allows an installation to specify a different SMFPRMxx parmlib member or to restart SMF if it terminates, the SETSMF operator command allows an installation to:

- Add a SUBPARM parameter value to those SMF parameter values already set for this IPL.
- Replace existing SMF parameter values (except ACTIVE, PROMPT, SID, and EXITS) with the specified parameters.

Parameters changed by SETSMF remain in effect only for the current IPL, or until replaced by a subsequent SET SMF or SETSMF command.

If the SMF parameter values set for this IPL include NOPROMPT, then the SETSMF command cannot be used to modify any of these values or to add the SUBPARM parameter value. For further information, see the description of the SMFPRMxx parmlib member later in this chapter.

Using SET SMS to Change SMS Parameters

The SET SMS command allows the installation to (1) select the IGDSMSxx parmlib member that the storage management subsystem (SMS) is to use or (2) dynamically replace the IGDSMSxx member that SMS is currently using. If you dynamically change the member, the effect last only for the duration of the IPL. For more information, see the description of the IGDSMSxx member later in this chapter.

Using SETSMS to Change SMS Parameters

The parameters on the SETSMS command are similar to the parameters in the IGDSMSxx parmlib member. Use the SETSMS command to:

- Activate an SMS configuration
- Change the active control data set (ACDS) that SMS is using
- Change the communications data set
- Change the synchronization interval
- Specify trace options for SMS

The effect of this command lasts only for the duration of the IPL; the IGDSMSxx member does not change. For more information, see the description of the IGDSMSxx member later in this chapter.

Using SET SLIP to Change SLIP Processing

The SET SLIP command selects an installation-supplied IEASLPxx parmlib member that contains the commands SLIP processing is to use. Using the IEASLPxx member to hold all of the SLIP commands may reduce contention for system resources and the amount of operator prompts during IPL.

Using SET PFK to Select the PFK Tables.

The SET PFK command selects the appropriate installation-supplied PFKTABxx member. This member contains the definitions for one or more program function key table(s) for a console or for a group of consoles. Using a PFKTABxx member provides automatic PF key initialization for the consoles as they are brought online. This processing reduces or eliminates the need for manually setting PF keys.

Starting GTF

When the operator starts the generalized trace facility (GTF), the parameters are obtained from the PARM field of the START command and from a parmlib member. If the operator issues START GTF, the IBM-supplied cataloged procedure (named GTF) is read. The PROC statement of that procedure names GTFPARM as the member from which GTF will get its parameters. If, however, the installation wants to substitute another member in place of GTFPARM, the operator may enter the alternate member name with the MEMBER keyword of the START command. (For further information on GTF initialization parameters, see the description of member GTFPARM later in this chapter. For other information on starting or using GTF, refer to the GTF chapter in *Service Aids*.)

Starting VLF,SUB = MSTR,NN = xx

The virtual lookaside facility (VLF) is a component that manages data objects in its own area of virtual storage. By managing the data objects in virtual storage, VLF can reduce the amount of I/O time required to retrieve the data.

VLF must be started after the master scheduler is initialized. The value for the keyword, NN, is the alphameric value appended to the VLF parmlib member, COFVLF.

Library Lookaside (LLA) Overview

Library lookaside (LLA) improves the performance of searching and fetching members from LNKLST and selected production libraries. LLA can also be used to control the versions of library members. The system obtains library directory entries from LLA's directory in storage, instead of making time consuming searches of partitioned data set (PDS) directories on DASD. The LLA directory can be dynamically refreshed to pick up new directory entries when DASD has been changed. LLA reduces the amount of fetch I/O by staging, or placing, copies of LLA-managed library modules in a data space managed by the virtual lookaside facility (VLF). When a staged module is requested, LLA retrieves it from virtual storage without I/O and with fewer processor instructions.

LLA can manage a data set in two ways:

FREEZE mode
BLDL/FIND uses LLA's version of the PDS directory.

NOFREEZE mode
BLDL/FIND uses the DASD version of the PDS directory.

By using the keywords FREEZE and NOFREEZE in a CSVLLAxx parmlib member, the installation can specify which way it wants LLA to manage the data set. LNKLST data sets accessed through the LNKLST concatenation will always be treated as FREEZE. For compatibility reasons, LLA-managed LNKLST data sets accessed outside of the LNKLST concatenation and LLA-managed non-LNKLST data sets default to NOFREEZE mode. However, for maximum performance benefit, LLA-managed data sets should be specified as FREEZE.

FREEZE enables an installation to take advantage of LLA's I/O reduction for directory search for the library and for load module fetch. With NOFREEZE, an installation may get faster performance by fetching the load modules from VLF instead of DASD. An installation can change the FREEZE or NOFREEZE mode of an LLA library at any time by using the MODIFY LLA command.

The FREEZE/NOFREEZE mode of an LLA-managed data set does not affect the staging of load modules into VLF. LLA maintains directory entries for all the data

sets it manages (FREEZE and NOFREEZE mode) and uses them when it stages load modules.

All LLA-managed data sets must be cataloged.

LLA specifies DISP=SHR on the libraries it manages. All jobs must specify DISP=SHR before submitting libraries to LLA's management. Jobs specifying DISP=OLD for an LLA-managed library will be enqueued until LLA is stopped or the library is removed from LLA.

Production load libraries that are accessed frequently and are seldom changed are good candidates to be managed by LLA.

Subsystems that manage directory entries for their own production libraries should carefully evaluate how their directory entry management process will interact with LLA. For example, if their directory entries are updated through BLDL, then LLA should manage the data set in NOFREEZE mode; otherwise, BLDL will always retrieve the LLA version of the directory entry. Such subsystems could benefit greatly from LLA's staging of modules.

The LLA directory should be refreshed whenever the subsystem's directory entries are refreshed to maximize the staging performance benefit.

If LLA terminates, either because of an error or as a result of a STOP LLA command, the operator can restart LLA by issuing the START LLA command.

Note: System performance is slowed if LLA terminates.

See the LNKLSTxx parmlib member (later in this chapter) for related information on the LNKLST concatenation.

See the CSVLLAxx parmlib member (later in this chapter) for related information on the syntax for CSVLLAxx.

Starting LLA

Before starting LLA, do the following:

1. Start VLF
2. Place the control statements needed to identify LLA's class of objects in the VLF parmlib member, COFVLFxx.

The SYS1.PARMLIB member needed to define LLA to VLF is in SYS1.SAMPLIB(COFVLFLL). This member defines a class of VLF objects, NAME(CSVLLA), with major name, EMAJ(LLA).

During initialization, the system issues the START LLA command from parmlib member IEACMD00. This command invokes the LLA procedure, and the system creates the LLA address space. The START LLA command identifies the CSVLLAxx parmlib member that LLA is to use to build the LLA directory, START LLA,LLA=xx. You can edit the LLA procedure in SYS1.PROCLIB so that LLA is started with START LLA,LLA=xx instead of START LLA. The control statements in the CSVLLAxx parmlib member identifies which data sets LLA is to manage and how LLA is to manage them. If LLA=xx is not specified, LLA

manages all the libraries in the LNKLST concatenation. When LLA=xx is specified, LLA manages the libraries in the LNKLST concatenation in addition to the list of libraries specified in the CSVLLAxx member. However, LLA will not manage a LNKLST library if it is specified with the REMOVE keyword in the CSVLLAxx member.

Modifying the LLA Directory

If updates have been made to the LLA libraries since the LLA directory was last built or refreshed, the operator can issue a MODIFY LLA command to activate the changes. The operator can issue a MODIFY LLA,REFRESH command to rebuild the entire LLA directory. To refresh a specific set of entry points and/or libraries in the directory, use the MODIFY LLA,UPDATE=xx command. See *System Commands* for the MODIFY LLA command. The UPDATE option identifies the CSVLLAxx member of SYS1.PARMLIB that contains the LLA control statements. The control statements specify the set of library member names, or libraries, to be selectively refreshed. See the CSVLLAxx parmlib member for information.

Whenever possible, use MODIFY LLA,UPDATE=xx rather than MODIFY LLA,REFRESH. When LLA directory entries are refreshed, LLA discards the associated staged module. This reduces LLA's performance benefit until LLA stages the modules again. Because LLA stages modules using its own directory entries, refresh LLA whenever a change is made to a LLA-managed data set to get the maximum performance benefit.

Also use MODIFY LLA,UPDATE=xx rather than stopping and restarting LLA. System performance is slowed anytime LLA is stopped.

The MODIFY LLA command does not reload (or refresh) modules that are already loaded, for example, modules in long-running or never-ending tasks. The refreshed version does not get picked up unless the module is loaded after the MODIFY LLA completes. To refresh such a module, you have two options:

- If the module has no corequisite requirement in LPALIB, you can use the subsystem's command to replace the module, or stop and then restart the long-running or never-ending task.
- Re-IPL the system with the CLPA option.

When an LLA library is specified with FREEZE, the user of that library always gets the LLA version of a member; thus, if the user makes an update to the member and then tries to browse it, the user sees the LLA version of the member. The LLA version does not have the changes in it until LLA is refreshed. If a user does multiple linkedits of a member in a FREEZE data set, the base for each subsequent linkedit does not include the previous linkedits; the base is the LLA version of the member. The recommended way to make updates in the production system is to use IEBCOPY under FREEZE mode. The member to be updated should be copied to another data set, the linkedits run against the second data set and then the updated member can be copied back to the LLA-managed data set. If LLA-managed production libraries must be updated directly, LLA should be refreshed to manage the data set in NOFREEZE mode.

If an LLA-managed library is compressed, LLA will provide obsolete directory entries. To avoid obsolete directory entries you must prevent access to the library modules that are being compressed. Do the following:

Issue a MODIFY LLA,UPDATE=xx command where the CSVLLAxx parmlib member includes a REMOVE statement identifying the data set being compressed

Do the compress

Issue a MODIFY LLA,UPDATE=xx command where the CSVLLAxx parmlib member includes a LIBRARIES statement to return the compressed library to LLA's management.

Notes:

Compressing LLA-managed data sets could cause abends, breaches of system integrity, and/or system failure if users try to use obsolete directory entries. Also, do not compress the library containing the IEBCOPY program used to perform the compress.

In a production system, use the MODIFY LLA command only after updates to the LLA-managed libraries are completed and verified. In a test system, you might want to update the LLA directory automatically by using JES to issue the MODIFY LLA command periodically.

Stopping LLA

The operator can issue a STOP LLA command to terminate LLA. Stopping LLA might be done when two or more systems share access to the same library directories and the directories must be modified simultaneously on each system. After stopping LLA and modifying the shared LLA-managed data sets as needed, the operator can use the START LLA command to restart LLA. System performance will be slowed until the operator restarts LLA.

An alternative to stopping LLA is to specify that the data set be removed from LLA by modifying LLA with a MODIFY LLA,UPDATE=xx command on each system. The CSVLLAxx parmlib member should contain the keyword REMOVE with the data set name that is to be modified. When the updates are complete, the operator can use the MODIFY LLA,UPDATE=xx command again with a CSVLLAxx parmlib member containing the keyword LIBRARIES with the data set name that was modified. The operator must issue the STOP and START or MODIFY commands on all of the systems sharing the LLA-managed data sets.

Using MODE to Change Machine Check Thresholds

The MODE command allows the operator to raise the threshold for specific types of machine checks. To improve the system's capability to recover from storage errors, the operator may wish to use this command to raise the thresholds of Instruction Processing Damage (PD) and System Recovery (SR) machine checks. (For additional information, see *Recovery and Reconfiguration.*)

Implicit System Parameters

Various system requirements, although not involving explicit parameters, affect the way the system performs. These system requirements may be considered as "implicit" parameters. They involve DD statements, data sets, hardware choices, and so forth. Some examples are:

- SYSABEND, SYSMDUMP, and SYSUDUMP DD statements. Without these statements, the parameters in parmlib members IEAABD00, IEADMR00, and IEADMP00 and the dump option lists in ABEND macro instructions are meaningless because an ABEND dump cannot be taken.
- The SMF data sets. If these data sets are not allocated on direct access volumes and cataloged, no SMF recording occurs.
- Addition of new modules to the LPALST concatenation through use of IEBCOPY or the Linkage Editor. Adding new modules affects the size and usefulness of the PLPA that is loaded by specifying the CLPA parameter at IPL.
- Choice of the device on which the PLPA paging data sets will reside. This choice affects the speed at which PLPA modules can be paged into real storage and thus influences system performance.
- Definition of page data sets by means of the DEFINE PAGESPACE command. The PAGE parameter, issued at IPL through parmlib and/or the operator, is meaningful only if the specified data sets have been previously formatted by the DEFINE PAGESPACE command. (See *Access Method Services Reference* for information on this command.)

Descriptions of Individual PARMLIB Members

The individual parmlib members, listed in alphabetic order, are described in the following topics.

Member Name: ADYSETxx

Use of the Member

ADYSETxx allows an installation to suppress dumps that it considers unnecessary because they duplicate previously taken dumps. ADYSETxx is used to suppress ABEND dumps that would be written to a SYSMDUMP data set (SYSMDUMPs) and SVC dumps, when the symptom data of a SYSMDUMP or an SVC dump duplicates the symptom data of a dump (of the same dump type) already taken. Dump analysis and elimination (DAE) uses the ADYSETxx parmlib member to determine the actions DAE is to perform.

DAE can take the following actions for both SYSMDUMPs and SVC dumps:

- Matching, which means that DAE compares each dump occurrence for a dump type (SVC dump or SYSMDUMP) to dumps previously recorded in SYS1.DAE.
- Suppressing, which means that DAE prevents a dump from being taken if the dump's symptom data duplicates the symptom data of a previous dump (of the same dump type) recorded in SYS1.DAE and if dump suppression is permitted.

For additional information on DAE, including how to create the SYS1.DAE data set, see *Dump and Trace Services*. Also, the "DAE" section of the *System Logic Library* contains a comprehensive discussion of matching and suppressing dumps, as well as other topics.

Parameter in IEASYSxx (or issued by the operator):

None

Syntax Rules

The following rules apply to the creation of ADYSETxx by means of the IEBUPDTE utility:

- Use an asterisk (*) in column one to indicate a comment record. Note that comment records cannot be continued and that comments cannot appear on a parameter record.
- DAE= must appear first on a parameter record, and START or STOP must appear on the record. (Specify one or the other; START and STOP are mutually exclusive parameters.)
- Use commas to separate parameters and to separate subparameters. If you use a parameter (such as RECORDS or SVCDUMP) that requires a subparameter, you must enclose the subparameter in parentheses.
- The format for an ADYSETxx record is:

```
DAE={START[,RECORDS(n)][,SVCDUMP(p)][,SYSMDUMP(p)]}
    {STOP                                          }
```

Syntax Examples:

```
DAE=START,RECORDS(900),SVCDUMP(MATCH,UPDATE)
DAE=SYSMDUMP(MATCH,UPDATE,SUPPRESS),START,RECORDS(600)
DAE=STOP
```

IBM-Supplied Defaults

IBM supplies three ADYSETxx parmlib members:

- ADYSET00 automatically starts DAE. It contains:

```
DAE=START,RECORDS(400),SVCDUMP(MATCH,SUPPRESS,UPDATE),
    SYSMDUMP(MATCH,UPDATE)
```

  Note: At system initialization, the ADYSET00 member is automatically in effect (DAE is active) because the IEACMD00 parmlib member contains the command SET DAE=00. If you do not want automatic activation of DAE, modify ADYSET00 to specify DAE=STOP.

- ADYSET01 allows the operator to stop DAE processing by issuing the command SET DAE=01. ADYSET01 contains:

```
DAE=STOP
```

- ADYSET02 allows the operator to start DAE processing by issuing the command SET DAE=02. ADYSET02 contains the same options as ADYSET00.

Internal Parameters

Parameter	Meaning and Use
MATCH	Specifies that DAE is to compare the symptoms from the current dump to those that have already been recorded in storage. (Coding MATCH does not imply that DAE will suppress duplicate dumps or update the SYS1.DAE data set.)
RECORDS(n)	Specifies the maximum number of symptom records to be placed in storage, where **n** is a decimal digit in the range 1 through 9999. Note that the system obtains records in multiples of 20. Therefore, if you specify RECORDS(23), the system will round up the number of records you specified to the next multiple of 20, and place 40 records in storage instead of the 23 specified. The default is 400 if you do not specify RECORDS(n).
START	Specifies that DAE is to be started. If already active, DAE is first stopped and then started again. **Note:** If you specify only DAE=START, the system does not issue an error message. However, without also specifying SVCDUMP(p) and/or SYSMDUMP(p), DAE does no dump processing and elimination.
STOP	Specifies that DAE is to be stopped and that all storage used by DAE is to be freed. This includes closing and unallocating SYS1.DAE.
SUPPRESS	Specifies that duplicate dumps are to be suppressed, if all other criteria for matching and suppressing dumps are met. (Coding SUPPRESS implies that that DAE will also match symptoms from the current dump to those already recorded; it does not imply that DAE will update the SYS1.DAE data set with any new symptoms.)
SVCDUMP(p)	Requests that DAE process SVCDUMPs, where **p** is a subparameter that specifies one or more of the following: MATCH, SUPPRESS, and/or UPDATE.
SYSMDUMP(p)	Requests that DAE process SYSMDUMPs, where **p** is a subparameter that specifies one or more of the following: MATCH, SUPPRESS, and/or UPDATE.
UPDATE	Specifies that SYS1.DAE is to be updated with the results of matching. The update can be a new record for a new set of symptoms or an increase to the count of occurrences for a duplicate set of symptoms. (Coding UPDATE implies that DAE will also match symptoms from the current dump to those already recorded; it does not imply that DAE will suppress any duplicate dumps.

Member Name: BLSCECT

Use of the Member

The BLSCECT member specifies the names of IBM and installation-supplied formatting exits for dump analysis. Interactive problem control system (IPCS) uses the BLSCECT parmlib member to build the IPCS exit control table (ECT). Each statement in BLSCECT specifies formatting exits and the context in which IPCS passes control to the exits. Issuing certain IPCS commands causes these exits to get control. IPCS accesses the BLSCECT parmlib member through the IPCSPARM DD statement.

If you allocate FILE(IPCSPARM), IPCS uses this file for all PARMLIB input. Otherwise, IPCS uses SYS1.PARMLIB.

For information on IPCS, see *IPCS User's Guide*. For information on IPCS commands, see *IPCS Commands*. For further information on formatting exits, see *IPCS Planning*.

The IMBED statement allows the description provided for BLSCECT to pertain to other members of SYS1.PARMLIB too. BLSCUSER is the name that IPCS reserves for you to use. You can use BLSCUSER to tailor IPCS without modifying any member distributed as part of any product. BLSCECT contains an IMBED statement specifying BLSCUSER.

Parameter in IEASYSxx (or issued by the operator):

None

If you allocate FILE(IPCSPARM), IPCS uses this file for all PARMLIB input. If you do not, IPCS uses SYS1.PARMLIB.

Syntax Rules

IPCS processes each statement as lines of TSO Command Language. See *TSO Commands* and *TSO User's Guide*.

IBM-Supplied Defaults

The default member is BLSCECT.

Internal Parameters

Statement	Parameter	Meaning and Use	Default Value
DATA		DATA statements associate specific types of AREAs or STRUCTURES with a scan routine, a find routine, a formatting routine or model, or a group of AREAs or STRUCTUREs. Use one DATA statement for each type of data specification.	
	AREA(*namelist*)	*(namelist)* is 1 to 31 alphanumeric characters identifying the names of the dump processing materials for IPCS and SNAP. The *namelist* must begin with an EBCDIC letter (A-Z) or ($, #, or @)	
	STRUCTURE(*namelist*)		
	FIND(*epname*)	FIND *(epname)* is 1 to 8 alphanumeric characters identifying the reentrant entry point name to locate the data types within this group based on a symbolic name. See the description of the GROUP statement. *epname* must begin with an EBCDIC letter (A-Z) or ($, #, or @)	
	FORMAT(*name*[,*level*])	FORMAT allows you to specify the *name* of the formatting routine to format the data types within this group. The *name* must begin with an EBCDIC letter (A-Z) or ($, #, or @) and may be followed by up to 7 letters, $, #, or @, or decimal digits (0-9). Lower case alphabetic characters are accepted and treated as upper case. The *level* can be specified as either JBB2125 or HBB3310. • JBB2125 indicates that the formatter does not support the extended interface introduced in MVS/SP3.1.0 (fmid HBB3310). The formatter is only given control when the block to be formatted can be described in terms acceptable to formatters written to support the MVS/SP2.1.2 (fmid JBB2125) interface. • HBB3310 indicates that the formatter supports the extended interface introduced in MVS/SP3.1.0.	HBB3310
	GROUP(*groupname*)	GROUP allows you to specify the name (*groupname*) of a control block group, such as RBs and UCBs. Pointers in other control blocks (such as TCBs) may be known to address one of these blocks, and IPCS may need to record the existence of the block in the dump directory before it is determined whether the block is, in fact, a PRB instead of another type of RB or a UCBTAPE instead of another type of UCB. The *groupname* specifies the type of control blocks to be associated with each type of data being defined by this DATA statement. The *name* must begin with an EBCDIC letter (A-Z) or ($, #, or @) and may be followed by up to 30 letters, $, #, or @, or decimal digits (0-9). Lower case alphabetic characters are accepted and treated as upper case. Do not reference a data type (AREA or STRUCTURE) as a *groupname* if it references another group data type. If an attempt is made to establish such a relationship, IPCS will detect an error, and the group data type associated with the explicitly-referenced group will be used instead.	

Statement	Parameter	Meaning and Use	Default Value
	MODEL(*modelname*)	Allows you to specify a *modelname* to format data types within this GROUP.	
		The *modelname* must begin with an EBCDIC letter (A-Z) or ($, #, or @) and may be followed by up to 7 letters, $, #, or @, or decimal digits (0-9). Lower case alphabetic characters are accepted and treated as upper case.	
	SCAN(*scanname*)	Allows you to specify a *scanname* to verify the instances of data types within this group.	
		The *scanname* must begin with an EBCDIC letter (A-Z) or ($, #, or @) and may be followed by up to 7 letters, $, #, or @, or decimal digits (0-9). Lower case alphabetic characters are accepted and treated as upper case.	
	ENVIRONMENT(IPCS) ENVIRONMENT(SNAP) ENVIRONMENT(ALL)	Specifies that the data requires support from IPCS, SNAP, or both.	IPCS
DIALOG		The DIALOG statement allows you to specify the component analysis dialogs that may be selected as part of the IPCS problem analysis dialog. The dialog may be selected from the component analysis panel.	
	NAME(*dialogname*)	*dialogname* specifies the name for the dialog. The *dialogname* must begin with an EBCDIC letter (A-Z) or $, #, or @ and may be followed by up to 7 letters, $, #, @, or decimal digits (0-9). Lower case alphabetic characters are accepted and treated as upper case.	
		Each *dialogname* must be unique. It must not duplicate the name of any other dialog or the EXIT EP(*epname*).	
	ABSTRACT(*text*)	Allows you to specify a 1-60 character line for the dialog as a quoted string. The character line can be upper and lower case characters. The displayed line will have the *dialogname* followed by the *text*	
	PARM('*text*')	Specifies the data passed to the ISPF SELECT service when the the dialog is active. Lower case characters are treated as upper case.	
		text must be entered in single quotes and cannot exceed 32,767 characters in length.	
END		The END statement terminates the processing of the current imbedded PARMLIB member accessed through BLSCECT. See the description of the IMBED statement. Any statement following an END statement in an imbedded PARMLIB member will not be processed. Processing of the member containing the IMBED statement continues. If a member contains a CLIST, the system considers that CLIST part of the member. If a CLIST generates and END statement, the CLIST generating the END statement, any other CLIST invoked by the currently imbedded member, and the currently imbedded member will terminate.	

Statement	Parameter	Meaning and Use	Default Value
EXIT	EP(*pgmname*)	Entry point name for the exit. *pgmname* may be 1-8 alphameric characters. The first character must be alphabetic (including $, #, or @.)	
	ABSTRACT(*text*)	Specifies a 1-60 character abstract for the exit as a quoted string. This causes the exit to be included on the IPCS dialog panel that enumerates component analysis exits. The format of the line displayed for the exit will be: *verb - text* If an exit is associated with multiple verbs, the first verb associated with the exit in BLSCECT is used. The system accepts upper and lower case alphabetic characters and will use both when the panel is formatted. Each *name* that is included on the IPCS dialog panel to enumerate the component analysis exits must be unique. It may not duplicate EXIT VERB(name) that is specified with an ABSTRACT and it may not duplicate the name of any other dialog.	
	AMASK X'00FFFFFF' X'7FFFFFFF'	Describes the logical AND mask. The exit routine uses the mask when it passes storage addresses to the dump access service. Any TSO INTEGER specification equivalent to these values is accepted. A TSO integer may be specified as decimal, hexadecimal, or binary. Specifying in hexadecimal is recommended. *Example:* EXIT EP(VTAMMAP) AMASK(X'00FFFFFF' This example is for VTAM.	7FFFFFFF (hex)
	ANALYZE	Specifies that the exit be invoked as a contention analysis exit. Contention analysis exits receive control when you issue the ANALYZE subcommand. The system invokes the ANALYZE exits in the order they are specified in BLSCECT. IPCS supports the ANALYZE exits. PRDMP ignores the ANALYZE exits. *Example:* EXIT EP(ISGDCONT) ANALYZE. This example will analyze the GRS ENQ contention.	
	ASCB	Specifies that the exit be invoked as an ASCB exit. The system invokes the ASCB exits in the order they are specified in BLSCECT. For more information, see *Service Aids*.	
	CBSTAT(name)	Specifies that the exit is a control block status exit for the type of control block specified by *name*. A valid *name* contains 1-31 characters. The first character must be alphabetic and the others may be alphanumeric. IPCS supports the CBSTAT exits. *Example:* EXIT EP(IEAVTRCA) CBSTAT(ASCB). This example is for a RTM ASCB status exit.	
	ENVIRONMENT(IPCS) ENVIRONMENT(ALL) ENVIRONMENT(SNAP)	Specifies that support for the exit is required in both the IPCS and SNAP environments or in only one of them.	IPCS

Statement	Parameter	Meaning and Use	Default Value
EXIT	FORMAT(name)	Specifies that the exit be invoked as a FORMAT exit. The order the FORMAT exits are listed in BLSCECT is the order in which they are invoked. To invoke the FORMAT exit, you can use the SUMMARY subcommand under IPCS, or you can specify the program name of the exit on the VERBEXIT subcommand . Do not code any parameters on the VERBEXIT subcommand.	
	HELP(*name*)	Specifies the *name* of the help panel that is to be displayed when a question mark is entered in the selection field on the exit selection line. The *name* must begin with an EBCDIC letter (A-Z) or $, #, @, and may be followed by up to 7 letters, $, #, @, or decimal digits (0-9). Lower case characters are treated as upper case.	
	PARM('*text*')	Specifies a parameter that is to be passed to the exit when it is selected from the panel that enumerates component analysis exits. The text must be enclosed in single quotes and may not be longer than 32,767 characters. Mixed upper and lower case alphabetic characters will be accepted and passed as is to a dialog.	
	TCB	Specifies that the exit be invoked as a TCB exit. The order the TCB exits are listed in BLSCECT is the order in which they are invoked. *Example:* EXIT EP(IEAVG701) CBSTAT(TCB). This example provides COMM TASK TCB exit for WTORs.	
	VERB(name)	Specifies a VERB name used with the exit identified with the IPCS VERBEXIT subcommand. If you code more than one VERB parameter using the same exit name, code separate EXIT statements for each VERB. *Example:* EXIT EP(IGVSFMAN) VERB(VLFDATA). The example provides an analysis of the VLF control blocks.	

Statement	Parameter	Meaning and Use	Default Value
IMBED	MEMBER(membername)	Specifies that PARMLIB (*membername*) should be processed before the processing of the current PARMLIB member is resumed. The *membername* must begin with an EBCDIC letter (A-Z) or ($, #, @) and may be followed by up to 7 letters, $, #, @, or decimal digits (0-9). Lower case alphabetic characters are accepted and treated as upper case. The IMBED statement allows you to imbed an installation-created parmlib member, BLSCUSER. BLSCUSER can contain imbeds for other parmlib members, and it can identify component and user-supplied formatting exits used for IPCS processing. The IBM-supplied BLSCECT member contains an imbed for BLSCECTX. BLSCECTX contains formatting exits for components other than the BCP. The IBM-supplied BLSECT contains IRAIPCSP which in turn contains formatting information for RSM. IRAIPCSP is a required IMBED statement.	
	REQUIRED	Specifies that IPCS will diagnose a failure to locate PARMLIB (*membername*) as an error. If you omit the REQUIRED parameter, IPCS accepts the failure to locate the member without comment.	
	ENVIRONMENT(IPCS) ENVIRONMENT(ALL) ENVIRONMENT(SNAP)	Specifies that support for the exit is required in both the IPCS and SNAP environments or in only one of them.	IPCS
NOTE		Use NOTE statements to transmit messages to the terminal, to FILE(IPCSPRNT), or to both. It is supported as a problem determination aid for construction and maintenance of PARMLIB members that may be reached via BLSCECT. NOTE statement syntax is identical to the syntax of the IPCS NOTE subcommand.	
SYMBOL		SYMBOL statements define groups of IPCS special symbols.	
	PREFIX(*prefixname*)	Specifies the initial characters, *prefixname* for a group of special symbols such as ASCBnnnnn, TCBnnnnnaaaaa, UCBxxxx, or ASTnnnnn. The *prefixname* must begin with an EBCDIC letter (A-Z) or $, #, or @ and may be followed by up to 29 letters, $, #, @, or decimal digits (0-9) to pass syntactic analysis. Lower case alphabetic characters are treated as upper case. Semantic analysis of the *prefixname* considers the type of SUFFIX specified. A *prefixname* must allow for the width of the associated suffix. SUFFIX(COUNT0 | COUNT1) - the prefix can be from 1 to 26 characters. SUFFIX(COUNT1NAME) - the prefix can be from 1 to 25 characters. SUFFIX(CPU) - the prefix can be from 1 to 29 characters. SUFFIX(DUALCOUNT) - the prefix can be from 1 to 21 characters. SUFFIX(NAME) - the prefix can be from 1 to 30 characters. SUFFIX(UNIT) - the prefix can be from 1 to 27 characters.	

Statement	Parameter	Meaning and Use	Default Value
SYMBOL	SUFFIX (COUNT0) | (COUNT1) | (COUNT1NAME) | (CPU) | (DUALCOUNT) | (NAME) | (UNIT)	Specifies the syntax requirements for the final characters for a group of special symbols. COUNT0 Specifies that the special symbols end with a number containing five decimal digits. Special symbols using a *COUNT0* suffix may have a prefix no longer than 26 characters. Specifies that the special symbols end with a number containing five decimal digits. Special symbols using a *COUNT1* suffix may have a prefix no longer than 26 characters. COUNT1NAME Specifies that the special symbols end with a number containing five decimal digits plus a name. The name begins with an alphabetic (A-Z) or $, @, #, and may contain additional characters that are alphabetic, $, @, #, or numeric (0-9). Special symbols using a *COUNT1NAME* suffix may have a prefix no longer than25 characters. CPU Specifies that the special symbols end with a number containing two decimal digits. Special symbols using a *CPU* suffix may have a prefix no longer than 29 characters. DUALCOUNT Specifies that the special symbols end with two numbers: Five decimal digits whose value falls between 1 and 65,535 Five alphabetic (base-26) digits whose value falls between "AAAAA" (0 in base-26) and "ZZZZZ" Special symbols using a DUALCOUNT suffix may have a prefix up to 21 characters.	

Statement	Parameter	Meaning and Use	Default Value
SYMBOL	SUFFIX	NAME specifies that the special symbols end with a name. The name must begin with one alphabetic (A-Z) or $, @, #. The second and subsequent characters, if any, may be alphabetic, $, @, #, or numeric (0-9) characters. Special symbols using a *NAME* suffix may have a prefix no longer than 31 characters minus the length of the prefix. UNIT Specifies that the special symbols end with a number containing four hexadecimal digits. Special symbols using a UNIT suffix may have a prefix no longer than 27 characters.	
	NAME(NAME)	Specifies the complete NAME of a supported symbol such as CVT GDA or PRIVATE. The name must begin with one alphabetic (A-Z) or $, @, # and may be followed by up to 30 letter, $, @, #, or decimal digits (0-9). Lower case alphabetic characters are accepted and treated as upper case.	

	STRUCTURE(*name*)
	AREA(*name*)
		Specifies that when the specified symbol(s) are entered, this data type is to be assumed.
		The *name* must begin with an EBCDIC letter (A-Z) or national character ($, #, @) and may be followed by up to 30 letters, national characters, or decimal digits (0-9). Lower case alphabetic characters will be accepted and treated as upper case.
		This allows an IPCS user to enter a shortened version of a command and to receive the same display that previously required longer command. For example, an IPCS user can enter
		`l cvt`
		and receive a display that the following command would provide:
		`l cvt structure(cvt)`	

TSO		The TSO statement allows you to invoke TSO commands and CLISTs during the processing of BLSCECT. TSO statement syntax is identical to the syntax of the IPCS TSO subcommand. The TSO statement is only required to invoke TSO commands whose names duplicate the names of IPCS PARMLIB statements.	

Member Name: CLOCKxx

Use of the Member

CLOCKxx performs two functions.

- Prompts the operator to initialize the TOD clock during NIP.
- Specifies the difference between the local time and Greenwich Mean Time (GMT).

Parameter in IEASYSxx (or entered by the operator):

CLOCK = { aa | (aa,bb...L) }

The two alphameric characters (aa, bb, and so forth) are appended to CLOCK to form the name of the CLOCKxx member of SYS1.PARMLIB. If you specify the L option in the syntax of the CLOCKxx member, or in reply to the 'SPECIFY SYSTEM PARAMETERS' message, the system writes all statements read from the CLOCKxx member to the operator's console.

Syntax Rules

The following rules apply to the creation of CLOCKxx by means of the IEBUPDTE utility:

1. Use columns 1 through 71. Do not use columns 72-80 for data; these columns are ignored.
2. Comments may be appear in columns 1-80 and must begin with "/*" and end with "*/"
3. One or more blanks may precede or follow the statement types.

An example of the syntax:

```
OPERATOR  NOPROMPT          /*System will prompt the operator only*/
                            /*if the TOD clock is not set */
TIMEZONE  W.04.00.00       /*The local time is West of GMT by 4 hours*/
```

IBM-Supplied Defaults

The IBM-supplied default member of SYS1.PARMLIB is CLOCK00.

Internal Parameters

Statement	Parameter	Meaning and Use	Default Value
OPERATOR	PROMPT	Specifies that the system is to prompt the operator during TOD initialization.	None
	NOPROMPT	Specifies that the system is not to prompt the operator during TOD initialization. If you specify NOPROMPT, the system prompts the operator to set the TOD clock only if the clock is not set.	None
TIMEZONE	d	Specifies the direction form Greenwich Mean Mean Time where d is either E for east of GMT or W for west of GMT.	W
	hh.mm.ss	Specifies the number of hours (hh) minutes (mm) and seconds (ss) that the local time differs from the GMT. The value for hh must be between 00 - 15. The value for mm.ss must be between 00 and 59. mm.ss values are optional.	00 00.00

Member Name: COFVLFxx

Use of the Member

The virtual lookaside facility (VLF) is an MVS/ESA component that enables an authorized program to store named objects in virtual storage managed by VLF and to retrieve these objects by name on behalf of users in multiple address spaces. VLF is designed primarily to improve performance by retrieving frequently-used objects from virtual storage rather than performing repetitive I/O operations from DASD.

A VLF class is a group of related objects made available to users through a component or application. To activate a class of VLF objects, VLF requires that a CLASS statement describing that group of objects be present in the active COFVLFxx parmlib member (the member named on the START command used for VLF).

For example, library lookaside (LLA) uses the class of VLF objects named CSVLLA. If the CLASS statement for CSVLLA is not included in the active COFVLFxx parmlib member, LLA cannot use VLF, and many of the performance and operational benefits of LLA will not be available.

IBM supplies a default VLF parmlib member (COFVLF00) that contains CLASS statements for the VLF classes used by IBM-supplied products. You might need to tailor some of these CLASS statements to meet your installation's needs. In addition, your installation can write applications that use VLF, and you must include the CLASS statements for those applications.

There are three items VLF requires for each VLF class used. They are:

1. The name of the class, specified on the required NAME parameter.

   IBM supplies the names of the classes it uses. These names start with the letters A-I.

2. The maximum amount of virtual storage that your installation wants VLF to use for the objects in the class.

   Unless you supply this value on the optional MAXVIRT parameter of the CLASS statement, VLF will use a default value. Generally, the information about the the IBM product that uses the VLF class provides some guidance about how to determine an appropriate value for MAXVIRT.

   For any given class, the goal is usually to provide an amount of virtual storage large enough to hold a working set of objects — those objects that are used frequently enough to justify keeping them in virtual storage to avoid DASD retrieval.

   When you specify the MAXVIRT value, make sure it is large enough to hold most or all of the frequently-used objects in a VLF class. An excessively small value tends to cause thrashing of the data in that VLF class, while an excessively large MAXVIRT value tends to increase the consumption of auxiliary storage because infrequently-used data is paged out, rather then discarded. Specifying MAXVIRT allows you to limit the maximum amount of auxiliary storage that could be used to back the VLF virtual storage that is holding the objects in the class.

The MAXVIRT value does not represent the *exact* amount of virtual storage that can be used to store objects. A small percentage of the storage is used for control information, and, in most cases, VLF begins to discard least recently used objects when the amount of virtual storage used for the class approaches a certain percentage of the MAXVIRT value. Therefore, allow some excess.

3. A list of the major names that represent the eligible sources of data for objects in the VLF class.

   How you specify the major names depends on whether the VLF class is a PDS class or a non-PDS class.

   For a PDS class, each major name identifies a unique partitioned data set and consists of a PDS name concatenated to the volume serial number. For a PDS class, use the EDSN and VOL parameters on the CLASS statement to define the major names.

   For a non-PDS class, the major name does not correspond to a partitioned data set. To specify the eligible major names for the class, use the EMAJ parameter on the CLASS statement. For an IBM-supplied class, use the product information to determine if anything other than the name(s) specified in the IBM-supplied default COFVLFxx member are eligible.

In addition to the class and major name, VLF also needs a minor name to identify a unique data object, but the minor names do not come from the COFVLFxx parmlib member.

Parameter in IEASYSxx (or issued by the operator):

None

Syntax Rules

The following syntax rules apply to COFVLFxx:

- The content of the member is one or more CLASS statements.
- A CLASS statement begins with the statement identifier "CLASS" and ends with the end of file (EOF) or when another class statement identifier is found. No explicit continuation syntax is required.
- Commas and blanks in any combination constitute a delimiter.
- Delimiters or comments can precede the statement identifier.
- Delimiters are not required between parameters with values; the right parenthesis after the specified parameter value is sufficient.
- Comments begin with /* and end with */. Comments are allowed between parameters.
- If a class statement contains duplicates of the following parameters, the system uses the first valid occurrence and issues a message that a duplicate parameter was specified:
  - NAME
  - MAXVIRT(nnn)
  - multiple VOL parameters for one EDSN parameter.

Syntax Format

```
CLASS     NAME(classname)
          {EDSN(dsn1) [VOL(vol)] EDSN(dsn2)...}
          {EMAJ(majname1) EMAJ(majname2)...}

          [MAXVIRT(nnn)]
```

Starting VLF

Issue the following command to start VLF:

```
START VLF,SUB=MSTR,NN=xx
```

The two alphameric characters (xx) are added to COFVLF to form the name of the COFVLFxx member. If you do not code NN=xx, the system defaults to COFVLF00.

Note that VLF will not start unless SUB=MSTR is specified on the START command. SUB=MSTR means that VLF can continue to run across a JES restart. It is recommended that you arrange for the VLF start command to be issued automatically during the IPL process.

Internal Parameters

Statement	Parameter	Meaning and Use	Default Value
CLASS		Each group of objects that VLF processes must have a CLASS statement defining it. The CLASS statement indicates that the following parameters define that particular group of objects to VLF.	
	NAME(*classname*)	NAME(*classname*) specifies the name of the VLF class. The *classname* may be one to seven alphanumeric characters including @, #, and $. IBM-supplied VLF class names begin with the letters A through I, for example, NAME(CSVLLA). Installation-supplied class names should begin with the letters J-Z or @, #, or $. NAME(*classname*) is required on the CLASS statement.	
	EDSN(*dsn*) [VOL(*vol*)]	For a PDS class, EDSN(*dsn*) identifies a partitioned data set name whose members are eligible to be the source for VLF objects in the class. The *dsn* can be 1 to 44 alphameric characters, including @, #, $, and periods (.).	
		You do not need to specify the volume if the cataloged data set is the desired one. If the data set is not cataloged, or if you have multiple data sets with the same name on different volumes, you must specify VOL(*vol*). Without the volume serial number, an un-cataloged data set is not included in the eligible data set name list. The system issues an informational message to the operator identifying any data sets that can not be allocated. The *vol* can be any combination of alphameric characters, including @, #, and $, or a dash (-).	
		Multiple occurrences of the same data set name with different volumes is acceptable. However, if duplicate entries of the same data set name and the same volume occur, the system issues an informational message and ignores the redundant information.	
		Do not use the EDSN parameter and the EMAJ parameter on the same CLASS statement.	
	EMAJ(*majname*)	EMAJ identifies an eligible major name (*majname*) for a non-PDS class, a class that does not have major names and minor names related to partitioned data sets and their members.	
		The *majname* can be 1 to 64 alphanumeric characters except comma (,), blank, or right parenthesis ( )), for example EMAJ(LLA).	
		Do not use the EMAJ parameter and the EDSN parameter on the same CLASS statement.	
	MAXVIRT(*nnn*)	MAXVIRT(*nnn*) is an optional parameter that allows you to specify, in 4K blocks, the maximum amount of virtual storage that VLF can use to store objects for the class.	4096 4K blocks
		The value for *n* must be a decimal number from 256 through 524288. You can specify up to six digits, including leading zeroes. The maximum value is the maximum data space size. If the system cannot obtain the amount of storage specified on MAXVIRT, it obtains as much as possible.	
		If the value for *n* is not in the valid range, the system uses 16 megabytes (4096 4K blocks.)	

Member Name: COMMNDxx

Use of the Member

COMMNDxx is an optional installation-created list of automatic commands the system internally issues as part of master scheduler initialization. COMMNDxx is useful for automatic entry of commands that are frequently issued at system initialization. You cannot use this member to issue JES commands, because JES is not started when the system issues the COMMNDxx commands. However, if an installation is running with JES3, and has defined consoles that will be actively managed by JES3, multiple console support (MCS) will initialize these consoles if they are online during IPL and defined to MCS. JES3 will be unable to obtain the device(s) and will prompt the operator for a device for each console. Including the following command in COMMNDxx avoids this and keeps the device online:

```
VARY  ddd, ONLINE
```

System trace is activated during the IPL. You can deactivate system trace or change trace options by using the TRACE operator command or by selecting a COMMNDxx parmlib member that contains the options you want. (See *System Commands* for information on the TRACE command.)

Note: Do not use COMMNDxx to issue SLIP commands. Instead, use the IEASLPxx parmlib member.

Some console-oriented commands affect the operation of a console. You can issue a console-oriented command directly from a console or, if the command contains a routing location operand (L=cc or L=cca) to indicate the console, you can place the command in a COMMNDxx member. If you place a console-oriented command (other than CONTROL M) in COMMNDxx and you do not specify a routing location operand, the system will not execute the command and might not generate an error message. The CONTROL M command does not support routing location operands, but it is valid in COMMNDxx.

When you specify the routing location operands, do so only as described in *System Commands*.

The COMMND00 member, if it exists, is read if CMD=xx is not included in the system parameter list (IEASYSxx) or is not specified by the operator. If initialization can't find either the specified COMMNDxx member or COMMND00, processing continues without automatic commands.

Parameter in IEASYSxx (or issued by the operator):

CMD = { aa | (aa,bb...) }

The two-character identifier (aa,bb,etc.) is appended to COMMND to identify the COMMNDxx member(s) of parmlib. Multiple members can be specified.

Notes:

1. Commands issued from COMMNDxx do not show on the console. Therefore, the results of these commands appear on the console without the operator's seeing the command.
2. Commands are issued in the order that they appear in COMMNDxx, but they are executed as follows:
   - Immediate commands, such as DISPLAY T, are executed sequentially as they are issued from COMMNDxx.
   - Execution of task-creating commands, such as DISPLAY A, is deferred until system initialization is complete. Then, factors such as multitasking, multiprocessing, and competition for resources influence the order in which these commands are executed. Thus, COMMNDxx should *not* be used to issue task-creating commands that must be executed in a specific order, because the execution order of these commands can vary.
3. A command placed in COMMNDxx must look exactly as it would if entered from the console. For example, to place the command SE 'TSO IS UP',CN=01 in COMMNDxx, specify the following:

```
COM='SE 'TSO IS UP',CN=01'
```

Syntax Rules

The following rules apply to the creation of COMMNDxx by means of the IEBUPDTE utility:

- Enter only one command per card image. Enter the COM=keyword, followed by the command enclosed in apostrophes. For example, to start TCAM through use of the IBM-supplied PROC, enter COM='S TCAM'.
- Do not specify continuation on any card image.

IBM-Supplied Defaults

None

Internal Parameters

Parameter	Meaning and Use	Default Value
COM = 'command name'	The specified command will be issued by the system during master scheduler initialization.	No commands will be issued.

Member Name: CONFIGxx

Use of the Member

CONFIGxx is a list of records that an installation can use to define a standard configuration of system elements. The system elements include the processors, the extended storage, Vector Facilities, storage, channel paths, devices, and volumes. You can use the configuration defined in CONFIGxx in two ways:

1. To compare the differences between the current configuration and the standard configuration as defined in a CONFIGxx member. When the operator issues the DISPLAY command with the M=CONFIG(xx) option, the system displays any differences.
2. To reconfigure some of the system elements. The operator can issue the CONFIG command with the MEMBER option.

Comparing the Current and Standard Configurations

In response to the DISPLAY M=CONFIG(xx) command, the system compares the contents of the CONFIGxx member to the existing configuration. It then displays the differences to the operator. The operator can then resolve these differences by using the CONFIG command.

Matching Configurations: If the existing configuration matches the one specified in CONFIGxx, the following message (IEE097I) is sent to the target console:

```
NO DEVIATION FROM REQUESTED CONFIGURATION
```

Nonmatching Configurations: If the existing configuration does not match the one specified in CONFIGxx, the following message (IEE097I) is sent to the target console:

```
syselm    DESIRED    ACTUAL
aaa       bbb        ccc
```

where:

syselm is the system element (CHP, CPU, DEVICE, ESTOR, STORAGE, VF, or VOLUME),

aaa is either the address of the system element or, for VF, is the

address of the CPU to which the Vector Facility is attached,

bbb is the status specified in CONFIGxx,

ccc is the existing status.

For example, if CPU 0 is actually offline but is specified as online in CONFIGxx, the message is:

```
CPU       DESIRED     ACTUAL
0         ONLINE      OFFLINE
```

Error in CONFIGxx Record: If a record in CONFIGxx is incorrect, the following message (IEE097I) is sent to the target console:

```
INVALID aaa SPECIFIED BEGINNING xxx
```

where:

aaa is either

REQUEST TYPE if the record is not recognized or
OPERAND if there is an error in specifying an operand.

xxx is the first sixteen characters of the invalid CONFIGxx record.

Reconfiguring System Elements

The CONFIG command with the MEMBER option enables the operator to reconfigure the system according the options in the specified CONFIGxx member. This reconfiguration is effective until the operator issues a different CONFIGxx MEMBER command or until the operator IPLs the system. With this command, the operator can reconfigure or verify the configuration of available channel paths, processors, real storage, real storage elements, extended storage elements, and Vector Facilities. These are defined by the CHP, CPU, STOR(E=id), ESTOR(E=id), and VF parameters in CONFIGxx. Other parameters found in CONFIGxx, such as DEV, VOL, and ESTOR(ddddM-ddddM), cannot be used for reconfiguration; they can be used only for verification.

See *System Commands* for more information on the DISPLAY and CONFIG commands.

Note: If you specify ONLINE and OFFLINE in the same CONFIGxx member, the ONLINE options are processed before the OFFLINE options. The system processes parameters that include the ONLINE option in the following order:

1. STOR
2. ESTOR
3. CPU
4. CHP

The system processes parameters that include the OFFLINE option in the following order:

1. CHP
2. CPU
3. ESTOR
4. STOR

Parameter in IEASYSxx:

None

Syntax Rules

Use the general syntax rules listed in the introduction to this chapter with the following exceptions:

- Continuation records are not permitted.
- Comment records are permitted and are indicated by an asterisk in column one.

IBM-Supplied Default

None

Internal Parameters

Parameter	Meaning and Use
CHP	Specifies the configuration of the channel paths. The syntax is as follows: CHP { xx / (xx-xx) / (ALL,id) / (list) } [,ONLINE / {,OFFLINE} [,UNCOND / ,FORCE]] where: **xx** is one or two hexadecimal digits that identify the channel path identifiers and indicate the different combinations of channel paths. **ALL,id** specifies all channel paths on the side identified by the side identifier (id), which can be either 0 or 1. Use CHP ALL,id only when your system is able to be partitioned. **list** can be any combination of the elements (except ALL,id) separated by a comma. The list must be enclosed in parentheses. **OFFLINE,FORCE** **CAUTION: FORCE is a very powerful option.** See the CONFIG command in *System Commands*. **Example:** `CHP (0,3-5,10,12-15)` This example specifies that channel paths 0, 3, 4, 5, 10, 12, 13, 14, and 15 are to be either verified or reconfigured as online, depending on the command issued.
CPU or CPUAD	Specifies the configuration for the processors. The parameter can be specified as CPU or CPUAD. The syntax is as follows: { CPU / CPUAD } { (x) / (x,x[,x]...) } [,ONLINE / ,OFFLINE] [,VFON / ,VFOFF] where x is a one hexadecimal digit identifying the processor address **Note:** If you specify VFON or VFOFF, you can specify only one processor address. **Example**: `CPU (2),ONLINE,VFON` Depending on the command issued, this example specifies that the processor addressed by 2 is to be either verified as online or reconfigured online and that the Vector Facility attached to the processor is to be either verified as online or reconfigured online.

Parameter	Meaning and Use
DEV or DEVICE	Specifies the configuration for devices.

The syntax is as follows:

$$\left\{\begin{array}{l}\text{DEV}\\\text{DEVICE}\end{array}\right\}\ \left\{\begin{array}{l}\text{ddd}\\\text{ddd-ddd}\\\text{(list)}\end{array}\right\}\ \left\{\begin{array}{l},\text{xx}\\,\text{(list)}\\,*\end{array}\right\}\ \left[\begin{array}{l}\underline{\text{,ONLINE}}\\,\text{OFFLINE}\end{array}\right]$$

where:

ddd specifies the device number (in hexadecimal) and reflects the different combinations of devices.

list can be any combination of the elements separated by a comma. The list must be enclosed in parentheses.

xx specifies the channel path(s) to be used to access the devices.

***** is used to verify that at least one channel path is online and can access the devices, regardless of which channel path it is.

The system displays a deviation message (IEE097I) for any of the following conditions:

- If channel paths are specified and all of the specified channel paths to the device(s) are not online or offline.
- If * is specified and there is no channel path online to access the device(s).
- If channel paths are not specified and * is not specified and all of the channel paths to the devices(s) are not online or offline.

Examples:

- To cause the system to verify that the devices with device numbers 334 and 33A can be accessed through channel path 12, code:

  ```
  DEV (334,33A),12,ONLINE
  ```

- To cause the system to verify that the devices with device numbers 340 through 34F can be accessed by all of their associated channel paths, code:

  ```
  DEV 340-34F,ONLINE
  ```

- To cause the system to verify that devices 100 and 200 through 21F can each be accessed by channel paths 01, 11, and 21, code:

  ```
  DEV (100,200-21F),(01,11,21)
  ```

  Note: This example, assumes the default of ONLINE.

- To cause the system to verify that devices 451, 453, and 455 can be accessed by at least one channel path, code:

  ```
  DEV (451,453,455),*,ONLINE
  ```

  Note: In this example, the * indicates that it is unimportant which channel path(s) are used to access the devices.

Parameter	Meaning and Use

ESTOR	Specifies the configuration for elements of extended storage. The syntax is as follows:
	ESTOR { (E=id) / (ddddM-ddddM) } [,ONLINE / ,OFFLINE]
	where:
	id is the identifier of an extended storage element. The identifier can be one to four hexadecimal digits from X'0000' to X'FFFF'.
	dddd is one to four decimal digits that must be a multiple of 4, followed by an M (megabytes). These values are the starting and ending addresses of the section of extended storage to be verified. You can specify only one range.
	Note: A CONFIGxx parmlib member that contains the ESTOR(ddddM-ddddM) can be used only as the target of a DISPLAY M=CONFIG command . It can not be used to reconfigure extended storage.
	Example:
	`ESTOR(E=03),ONLINE`
	This example indicates that a section of extended storage, whose identifier is X'03', is to be either verified or reconfigured as online.
	Example:
	`ESTOR (0M-64M),ONLINE`
	This example indicates that a section of extended storage (location 0 through location 67,108,864) is to be verified as online.

STOR or STORAGE	Specifies the configuration for sections of real storage. Multiple ranges can be specified. The parameter can be specified as STOR or STORAGE. The syntax is as follows: { STOR / STORAGE } { ddddddK-ddddddK / xxxxxxxx-xxxxxxxx / ddddM-ddddM / (E=id) / (list) } [,ONLINE / ,OFFLINE] where: **ddddddd** is one to seven decimal digits, followed by a K, which are the starting and ending addresses of the section. Each address represents a multiple of 1024 bytes. If necessary, the system will round the low address down to the next lower 4K boundary and the high address up to the next higher 4K boundary. (This rounding is done to begin and end a section of storage on a 4K boundary.) Note that the system does not reconfigure a section of real storage (in response to a CONFIG command) when the section is specified in this manner. **xxxxxxxx** is one to eight hexadecimal digits that address the first and last bytes of the section. If necessary, the system will round the low address down to the next lower 4K boundary and the high address up to the next higher 4K boundary. (This rounding is done to begin and end a section of storage on a 4K boundary.) Note that the system does not reconfigure a section of real storage (in response to a CONFIG command) when the section is specified in this manner. **dddd** is one to four decimal digits, followed by M, which are starting and ending addresses of the section. Each address represents a multiple of one megabyte (1,048,576 bytes). **E=id** specifies a storage element identified by the storage element identifier (id), which is one hexadecimal digit. **list** can be any combination of the elements (except E=id) separated by a comma. The list must be enclosed in parentheses. **Example**: `STOR 8192K-32768K,ONLINE` This example indicates that a section of real storage (location 8,388,608 through location 33,554,431) is to be verified as online.

Parameter	Meaning and Use
VF	Specifies the configuration for a Vector Facility and the online processor to which it is attached. The syntax is as follows: VF(x) { ,ONLINE / ,OFFLINE } where x is a one-digit processor address (in hexadecimal) that represents the processor to which the Vector Facility is attached. **Example**: VF(2),OFFLINE Depending on the command issued, this example specifies that the Vector Facility attached to the processor addressed by 2 is to be either verified as offline or reconfigured offline.
VOL or VOLUME	Specifies the configuration for DASD volumes. The parameter can be specified as VOL or VOLUME. The device number on which a volume should be mounted can optionally be specified. The syntax is as follows: { VOL / VOLUME } { v / v=ddd / list } where: **v** is the volume name. **ddd** is the device number. Any digit in the three digit device number can be replaced with an 'X' to indicate that the value of that digit is not important. **list** can be any combination of the elements separated by a comma. **Example**: `VOL PAGE1,P00045=3X5` This example specifies that PAGE1 and P00045 are to be verified as mounted; P00045 should be mounted on a device with a device number in the range 305 through 3F5.

Member Name: CONSOLxx

Use of the Member

CONSOLxx is an installation-created member of SYS1.PARMLIB in which you can define a console configuration to meet the particular needs of your installation.

In CONSOLxx, you define multiple console support (MCS) consoles. You define the specific characteristics of the master console and for up to 98 secondary consoles. While the master console is the principal means of communicating with the system, the secondary consoles often serve specialized functions. You can define the characteristics of any specific console to meet the needs of your installation. You might need to have PF keys on one console issue one group of system commands while on another console you may need the PF keys set to issue a different set of commands. You might need to have particular messages routed to one console and you might want those messages deleted in a certain way. You might need a cluster of consoles to serve different functions for a subsystem. One console in the cluster may display, in a particular format, only the messages associated with a particular group of routing codes. Through CONSOLxx, you can specify the configuration and have the system automatically initialize the consoles.

Through CONSOLxx, you can specify initialization values for all console configurations. For example, you can define the characteristics of the hardcopy log and set routing codes for messages that do not have routing information.

CONSOLxx also provides a way to centralize the definitions of the console configuration for your installation. Within CONSOLxx, you specify the MPFLSTxx member of parmlib for message processing control. You also specify the PFKTABxx member of parmlib that defines any PFK tables you require. Specifying the MPFLSTxx member and the PFKTABxx member within the CONSOLxx member makes it easier to maintain the three related members of parmlib. See the descriptions of the MPFLSTxx and PFKTABxx parmlib members.

CONSOLxx is closely related to the IODEVICE statements used in the input stream to the MVS configuration program. The device number you specify in CONSOLxx must match the device number on an IODEVICE statement. See *MVSCP Guide*.

The system **requires** that a CONSOLxx member be identified at IPL. You can use the CON parameter in the IEASYSxx member of parmlib, or you can select the CONSOLxx member (CON=xx) when responding to the prompt for system parameters.

Through the CONTROL, SET, and VARY commands, you can change some of the characteristics specified in CONSOLxx. However, these changes are temporary; they last only for the current IPL. At the next IPL, the definitions in the CONSOLxx member are in effect. See *System Commands* for more information.

CONSOLxx contains four statement types:

- CONSOLE
- INIT
- HARDCOPY
- DEFAULT

CONSOLE Statement

On the CONSOLE statement, you identify the console by the device number and unit type and then define the characteristics for that console. For each console, you define the following characteristics:

- Command group
- Operating mode
- An alternate console
- Routing codes
- Message deletion
- Display area specifications
- Out-of-line display areas
- The PFK table name
- Message routing instructions
- Monitor specifications

You must have a CONSOLE statement for each device number. You must define the device number on the IODEVICE statement for MVSCP before you can use it on a CONSOLE statement. The device number and the unit type on the IODEVICE statement for MVSCP and on the CONSOLE statement for CONSOLxx must match. A CONSOLxx member can include multiple CONSOLE statements; one for each console in the configuration.

INIT Statement

On the INIT statement, you can define the initialization values for the communications task. You can:

- Specify the limits for WTO and WTOR buffers
- Specify the MPFLSTxx member to use with this CONSOLxx member.
- Specify the PFKTABxx member to use with this CONSOLxx member.
- Activate the action message retention facility
- Activate the WTO user exit IEAVMXIT
- Specify how the MONITOR command is to display mount and demount messages.
- Specify an MVS command delimiter to allow an operator to issue multiple commands concurrently.

The INIT statement is optional. However, if you code an INIT statement, you can have only one in a CONSOLxx member.

DEFAULT Statement

The DEFAULT statement allows you to define the default routing codes for all write-to-operator (WTO) and write-to-operator-response (WTOR) messages that do not have routing codes specifically assigned.

The DEFAULT statement is optional. If you do code a DEFAULT statement, you can code only one in a CONSOLxx member. If you do not code the DEFAULT statement, the system assigns routing codes 1 through 16 to the messages.

HARDCOPY Statement

With the HARDCOPY statement, you define the characteristics for the hardcopy log:

- Whether it is a device or SYSLOG,
- The routing codes of the messages it is to receive
- The kinds of commands (commands, responses or status display) it is to record

The HARDCOPY statement is optional. If you code a HARDCOPY statement, you can have only one in a CONSOLxx member. If you do not code a HARDCOPY statement, the system uses SYSLOG.

Parameter in IEASYSxx:

CON= { **aa** / **(aa,L)** }

The CON parameter is required for IPL. The two alphameric characters (aa) are appended to CONSOL to form the name of the CONSOLxx member of SYS1.PARMLIB. If you specify the L option in IEASYSxx, or in reply to the 'SPECIFY SYSTEM PARAMETERS' message, the system writes all of the statements read from the CONSOLxx member to the operator's console.

Syntax Rules

Use the general syntax rules listed in the introduction to this chapter with the following exceptions and additions:

- You may define up to 99 consoles within a CONSOLxx member.

  Note: The composite consoles count as two consoles.

- On each CONSOLE statement, DEVNUM is required and must be the first parameter.
- Data, including comments, must be contained in columns 1-71; the system ignores columns 72-80.
- Comments must begin with "/*" and end with "*/"
- You do not need to code delimiters between parameters.
- One or more blanks may precede or follow the statement types.
- Parameter values must be set off by parentheses. If you code multiple values on certain keywords, separate the values with a blank or a comma.
- Do not use blanks in the middle of a keyword, in the middle of a value, or between the parameter and the left parentheses before the value.
- A statement type must be the first data item on a record.
- You may code comments before any statement type.
- A statement type continues to the next statement type in the member or until the end of the member.

IBM-Supplied Default

None

Syntax Format for a CONSOLE statement

```
CONSOLE   DEVNUM {(devnum[,devnum])}
                 {(SUBSYSTEM)      }

          UNIT {(unittype)}
               {(PRT)     }
               {(COMP)    }

          AUTH {(MASTER)              }
               {(INFO)                }
               {([SYS][,IO][,CONS])   }
               {(ALL)                 }

          USE {(FC)}
              {(MS)}
              {(SD)}

          ALTERNATE(devnum[,devnum])

          ROUTCODE {(ALL)                  }
                   {(NONE)                 }
                   {(nnn[,nnn-nnn][,nnn]...)}

          LEVEL {(ALL)                        }
                {([ALL][,NB])                 }
                {([R][,I][,CE][,E][,IN][,NB]) }

          CON {(Y)}
              {(N)}

          SEG(nn)

          DEL {(Y) }
              {(R) }
              {(RD)}
              {(N) }

          RNUM {(nn)}
               {(5) }

          RTME {(nnn)}
               {(2)  }

          MFORM {(M)          }
                {([J][,S][,T])}

          AREA {(nn[,nn]...)}
               {(NONE)      }

          PFKTAB(tablname)

          MSGRT(['inst'][,'inst']...)

          MONITOR([JOBNAMES[-T]][,SESS[-T]][,STATUS])

          UTME {(nnn)}
               {(30) }
```

Syntax Format for an INIT statement

```
INIT      MLIM   {(nnnn)  }
                 {(1500)  }

          RLIM   {(nnn)   }
                 {(10)    }

          AMRF   {(Y)     }
                 {(N)     }

          UEXIT  {(Y)     }
                 {(N)     }

          MPF    {(NO)    }
                 {(nn)    }

          PFK    {(NONE)  }
                 {(nn)    }

          MONITOR([SPACE][,DSNAME])

          CMDDELIM {(c)     }
                   {(X'hh') }
```

Syntax Format for a HARDCOPY statement

```
HARDCOPY  DEVNUM   {(devnum[,devnum])}
                   {(SYSLOG)         }

          ROUTCODE {(ALL)                    }
                   {(NONE)                   }
                   {(nn[,nnn-nnn][,nnn]...)  }

          CMDLEVEL {(NOCMDS)}
                   {(INCMDS)}
                   {(TCMDS) }
                   {(CMDS)  }
```

Syntax Format for a DEFAULT statement

```
DEFAULT   ROUTCODE {(ALL)                    }
                   {(NONE)                   }
                   {(nnn[,nnn-nnn][,nnn]...) }
```

Internal Parameters

Statement	Keyword(Value)	Meaning and Use
CONSOLE		CONSOLE indicates the beginning of a statement that defines the characteristics of a console.
	DEVNUM {(devnum) (devnum,devnum) (SUBSYSTEM)}	DEVNUM specifies the device number of the console. DEVNUM is **required** and must be the first keyword on the CONSOLE statement. *devnum* is three hexadecimal digits (X'000-FFF') and **must** be the same device number as the one specified for the device on the IODEVICE statement in the MVS configuration program. *(devnum,devnum)* indicates the device is a composite console: the first devnum is the input part of the console. The second devnum is the output part of the console. The input or output part of a composite console can not be defined again as a single device or as part of another console, for example, with a different companion device. If you code (devnum,devnum), you must specify UNIT(COMP) and there must be two IODEVICE statements for this console in the input stream for the MVS configuration program. The composite consoles are treated as two consoles. *(SUBSYSTEM)* indicates that this console is reserved for a subsystem, such as JES3 or OCCF. **Note:** The only keyword that is valid with DEVNUM(SUBSYSTEM) is AUTH. AUTH must have a value other than MASTER; the subsystem console must not be the master console.
	UNIT {(2740) (3270-X) (3277-2) (3278-2) (3278-2A) (3278-3) (3278-4) (3279-2A) (3279-2B) (3279-2C) (3279-3A) (3279-3B) (PRT) (COMP)}	UNIT specifies the unit type of the console. The unit type must be a valid console device listed here. See the end of the CONSOLxx discussion for further information. *PRT* specifies the console is a printer. The IODEVICE statement in MVSCP must specify a valid device type that can be defined as a printer. *COMP* identifies a composite console. See the chart of devices that can be used as consoles. With UNIT(COMP), you must also specify DEVNUM(devnum,devnum), and there must be two IODEVICE statements for this device in the MVSCP input stream. The first devnum, the input console, corresponds to a reader; the second devnum, the output console, and corresponds to the printer. **Notes:** 1. Specify 3270-X only for the 3278s and 3279s connected to a controller that supports the Read Partition Query feature. The chart at the end of this section contains information on devices used as consoles. 2. If you do not code the UNIT keyword, the system uses the related IODEVICE statement for the device number to determine the unit type. If the IODEVICE statement indicates that it is a display device, the system will default the UNIT value to 3270-X.

Statement	Keyword(Value)	Meaning and Use
	AUTH {(MASTER) (INFO) ([SYS][,IO][,CONS]) (ALL)}	AUTH specifies the group of operator commands that can be entered from the console. *MASTER* indicates that this is the master console. From the master console, you can enter all MVS operator commands. You can have only one master console; therefore, code AUTH(MASTER) on only one CONSOLE statement. **Note:** Do not code DEVNUM(SUBSYSTEM) AUTH(MASTER) on the CONSOLE statement. The master console can not be a subsystem console. The corresponding authority levels for JES3 are: MASTER: JES3 authority level = 15 CONS: JES3 authority level = 10 I/O: JES3 authority level = 10 SYS: JES3 authority level = 5 INFO: JES3 authority level = 0 *INFO* specifies that any informational commands can be entered from this console. INFO is the default. *SYS* specifies that system control commands and informational commands may be entered from this console. *IO* specifies that I/O control commands and informational commands may be entered from this console. *CONS* specifies that console control commands and informational commands may be entered from this console. ALL specifies that information, system control, I/O control, and console control commands may be entered from this console. **Note:** AUTH is not valid with UNIT(PRT).
	USE {(FC) (MS) (SD)}	USE specifies the operating mode of the console. *FC* indicates full-capability. If the console is a display device, a 2740, or a composite, specify USE(FC). If you do not specify USE, FC is the default if the console is a display device, a 2740, or a composite *MS* indicates message stream mode. If the console is a printer, specify USE(MS). If you do not specify USE, MS is automatically the default if the console is a printer. *SD* indicates the console is to be in status display mode.
	ALTERNATE {(devnum) (devnum,devnum)}	ALTERNATE specifies the device number of the alternate console. The alternate must be defined on a different CONSOLE statement. **Note:** An output only console can have any console as its alternate. A full-capability console must have as its alternate a console defined as full-capability or one that can become full-capability. A printer can not be the alternate for a full-capability console. *devnum* must be three hexadecimal digits (X'000-FFF'). *devnum,devnum* indicate that the alternate is to be a composite console. **Note:** A composite console can not be specified as a single device or as part of another composite console with a different companion device. However, the same combination may be specified as an alternate on another CONSOLE statement. If you do not code ALTERNATE, the master console is the alternate.

Statement	Keyword(Value)	Meaning and Use
	ROUTCODE {(ALL) (NONE) (nnn[,nnn-nnn][,nnn]...)}	ROUTCODE specifies the routing codes assigned to the console. ALL specifies all routing codes, 1 through 128. nnn specifies a decimal value from 1 through 128. nnn-nnn specifies a range of decimal value with the lower value first. NONE is the default. For a master console NONE indicates routing codes 1 and 2. For all other consoles, NONE indicates no routing codes. See *Routing and Descriptor Codes* for more information of the codes and their usage.
	LEVEL {(ALL) [ALL][,NB] [R][,I][,CE][,E][,IN][,NB]}	LEVEL specifies the message levels for the console. ALL is the default. *ALL* indicates that the console is to receive all messages. *NB* This console is to receive no broadcast messages. *R* This console is to receive the WTOR messages. *I* This console is to receive immediate action messages. *CE* This console is to receive critical eventual action messages. *E* This console is to receive eventual action messages. *IN* This console is to receive informational messages.
	CON{(Y) (N)}	CON indicates whether the console is to function in conversational or nonconversational mode. *Y* indicates conversational mode; on this console you must verify all messages selected for message deletion with the cursor, or selector pen or through the CONTROL command. **Note:** CON is not valid when UNIT is 2740, PRT or COMP. *N* indicates nonconversational mode; all messages selected for deletion are automatically deleted.
	SEG (nn)	SEG specifies the number of lines in the message area that can be deleted when the CONTROL E, SEG command is entered. *nn* is a decimal value from 1 to the number of lines in the message area. For default values for SEG, see the chart at the end of this section. **Note:** SEG is not valid when UNIT is 2740, PRT or COMP.
	DEL {(Y) (R) (RD) (N)}	DEL specifies the message deletion mode of the console. *Y* indicates that all messages selected for deletion are deleted when the screen becomes full. *R* indicates roll mode. In roll mode, the system deletes a specified number of messages from the screen when a time interval elapses. Deletion occurs only if the screen is full and messages are waiting to be displayed. *RD* specifies roll mode except for messages awaiting actions. These messages are displayed at the top of the screen. RD is the default. *N* indicates that no automatic message deletion is in effect. Messages must be deleted manually. **Note:** DEL is not valid when UNIT is 2740, PRT, or COMP.

Statement	Keyword(Value)	Meaning and Use
	RNUM (nn)	RNUM specifies the maximum number of lines included in one message roll. *nn* is a decimal value from 1 to the number of lines in the message area. The ***default*** value for all unit types is 5 line. **Note:** RNUM is not valid when UNIT is 2740, PRT, or COMP.
	RTME (nnn)	RTME specifies the number of seconds between message rolls. *nnn* is a decimal value from 1 to 999. The ***default*** value for all unit types is 2 seconds. **Note:** RTME is not valid when UNIT is 2740, PRT or COMP.
	MFORM { (M) / ([J][,S[[,T] }	MFORM specifies the display format of the messages. *M* indicates that the system is to display only the text of each message. The message display does not include time stamp, job ID, or job name information, or the system name. M is the default. *J* specifies that the display is to include the job ID or name. *S* specifies that the display is to include the name of the system originating the message. *T* specifies that the display is to include a time stamp.
	AREA { (nn[,nn]...) / (NONE) }	AREA specifies the size(s) of the out-of-line display area(s) on the display console. *nn* is a decimal specifying the number of lines in one display area. The first number defines the bottom area of the screen. Subsequent numbers define areas working toward the top of the screen. The minimum number of lines in an area is 4. The maximum number of areas is 11. The sum of the lines in all of the areas must not exceed the screen size. For screen sizes and maximum and default values for AREA. See the chart at the end of this section. **Note:** AREA is not valid when UNIT is 2740, PRT or COMP.
	PFKTAB (tablname)	PFKTAB specifies the name of the table that defines the PFK definitions. *tablname* identifies the 1-8 alphameric name for the table and must be the same as one defined in a PFKTABxx member. The PFKTABxx member must be active, that is identified in the PFK parameter on the INIT statement in CONSOLxx. See the description of the PFKTABxx parmlib member. **Note:** PFKTAB is not valid when UNIT is 2740, PRT, or COMP.
	MSGRT(['inst'][,]['inst']...)	MSGRT allows you to specifiy the routing instructions to direct routable system displays to a specified message area, console or both. *inst* can be one routing instruction or a string of routing instructions. Each instruction must be enclosed in single quotes and must be less than 123 characters. See the description of the MSGRT command in *System Commands* for the syntax of routing instructions. **Note:** MSGRT is not valid when UNIT is PRT.
	MONITOR ([JOBNAMES[-T]][,SESS[-T]] [,STATUS])	MONITOR allows you to specify how selected events will be monitored. *JOBNAMES* specifies that each job name will be displayed when the job starts and ends. Unit record allocation will be displayed when the step starts. If a job terminates abnormally, the job name will appear in a diagnostic message. *SESS* specifies that the user identifier for each time sharing terminal is to be displayed when the session starts and ends. If a session terminates abnormally, the user identifier will appear in a diagnostic message. [-T] specifies the time is to be displayed with the job name or the user identifier. The time is in the format hh:mm:ss.
	UTME (nnn)	UTME specifies the time interval in seconds for updating dynamic displays. *nnn* is a decimal value from 10 to 999. The default value for nnn is 30 seconds. **Note:** UTME is not valid when UNIT is 2740, PRT, or COMP.

Statement	Keyword(Value)	Meaning and Use
INIT		Specifies that the communication tasks initialization values follow.
	MLIM (nnnn)	MLIM specifies the maximum number of buffers that the write to operator (WTO) messages can use. *nnnn* is a decimal value between 20 and 9999. The ***default*** is 1500. **Note:** In a JES3 system, the MLIM value must be less than the number of JES3 staging areas defined for that processor.
	RLIM (nnn)	RLIM specifies the maximum number of buffers used by the write-to-operator-with-reply (WTOR) routines. *nnn* is a decimal value between 5 and 100. The ***default*** is 10.
	AMRF { (Y) / (N) }	AMRF specifies whether the action message retention facility is to be active or not. *Y* indicates that it is to be active, and *N* indicates that it is not to be active. The default (Y) is active.
	UEXIT { (Y) / (N) }	UEXIT specifies whether the WTO user exit IEAVMXIT is to be active or not. *Y* indicates that it is to be active and *N* indicates that it is not to be active. The default (Y) is active.
	MPF { (NO) / (nn) }	MPF indicates whether a message processing facility is to be active or not. No is the default; the facility is not active. *nn* must be two alphameric characters indicating the MPFLSTnn member to use.
	PFK { (NONE) / (xx) }	PFK specifies the PFKTABxx member in parmlib that contains any PFK tables for your consoles. None indicates that no PFK tables are defined; the system is to use the IBM-supplied default PFK definitions that are described in *System Commands xx* must be two alphameric characters.
	MONITOR ([SPACE][,DSNAME])	MONITOR specifies how the system is to display certain information. *SPACE* indicates that, in demount messages, the system is to display the available space on the direct access volume. *DSNAME* indicates that, in mount messages, the system is to display the name of the first non-temporary data set allocated on the volume to which a message refers. **Note:** If a data set has a disposition of delete, its name does not appear in the mount messages.
	CMDDELIM { (c) / (X'hh') }	CMDDELIM specifies an MVS command delimiter that the installation chooses. CMDDELIM allows the operator to enter multiple commands on the console at one time. Each command separated by the delimiter is processed separately. Do not use the command delimiter within a quoted string. If you do, the system considers the delimiter as part of the text and it will not process the delimiter. The order in which multiple commands the system executes the commands may be different from the order in which the operator entered them. PF keys continue to use a semicolon (;) as the delimiter. Choose the command delimiter carefully. It must not conflict with any character defined by any subsystem in use. The command delimiter may be specified in EBCDIC (c) or in hexadecimal (X'hh') format. Valid values are

EBCDIC	HEXADECIMAL
<	4C
>	6E
&	50
;	5E
%	6C
?	6F
:	7A
"	7F

Statement	Keyword(Value)	Meaning and Use
DEFAULT		DEFAULT allows you to define the default routing codes for all write to operator (WTO and WTOR) messages that do not have routing codes specifically assigned.
	ROUTCODE {(ALL) (NONE) (nnn[,nnn-nnn][nnn]...)}	ROUTCODE specifies the default routing code(s) for messages that do not have any routing information. *ALL* specifies 1 through 128 are to be assigned. *NONE* specifies that no routing codes are to be assigned. *nnn* is a decimal number from 1 to 128. You can specify a range of routing codes by coding an ascending range of numbers. If you do not code ROUTCODE, the ***default*** is 1 through 16.
HARDCOPY		HARDCOPY identifies any hardcopy device and defines what messages and commands responses are to be recorded.
	DEVNUM {(devnum) (devnum,devnum) (SYSLOG)}	DEVNUM specifies the output device. *devnum* specifies the device number of a console that is to be the hardcopy log device. devnum is three hexadecimal digits ranging from X'000-FFF'. *(devnum,devnum)* indicates the device is a composite console. *(SYSLOG)* indicates that the data going to the hardcopy log will go to the system log. If you do not code a HARDCOPY statement or if you do not code DEVNUM on a HARDCOPY statement the default is SYSLOG. If the hardcopy console is not usable, the system defaults to SYSLOG. **Note:** A display console can not be a hardcopy log console. The device you specify for the hardcopy log console must be defined as a console in CONSOLxx.
	ROUTCODE {(ALL) NONE) (nnn[,nnn-nnn][nnn]...)}	ROUTCODE specifies the routing codes the hardcopy log is to receive. In addition to the routing codes you specify, the hardcopy log also receives the minimum set (1,2,3,4,7,8,10 and 42). *ALL* indicates that the hardcopy log is to receive all routing codes (1-128). ALL is the default. *NONE* indicates that the hardcopy log is to receive no routing codes. It will receive the minimum set (1,2,3,4,7,8,10,42). *nnn* is a decimal number from 1 to 128. You can specify a range of routing codes by coding an ascending range of numbers.
	CMDLEVEL {(NOCMDS) (INCMDS) (STCMDS) (CMDS)}	CMDLEVEL specifies the type of commands that are to be recorded on the hardcopy log. *NOCMDS* indicates that no operator or system commands or responses are to go to the hardcopy log. *INCMDS* specifies that operator and system commands and responses are to go to the hardcopy log but no status displays. *STCMDS* specifies that operator and system commands, responses, and static status displays are to go to the hardcopy log but no time-interval updated status displays. *CMDS* specifies that operator and system commands, responses, and status displays, including static and time-interval updated status displays, are to go to the hardcopy log.

DEVICES USED AS MCS CONSOLES

Device	Input/Output Devices	Composite Input Devices	Composite Output Devices	Output Only Devices	Specified As	Description
2501		X				Reader
2540		X				Reader
3505		X				Reader
3525-5		X				Punch
1403			X	X		Printer
3203-5			X	X		Printer
3211			X	X		Printer
3284-1			X	X		Printer
3284-2			X	X		Printer
3286-1			X	X		Printer
3286-2			X	X		Printer
3270-X	X			X		Display
3277-2	X			X		Display
3278-2	X			X		Display
3278-2A	X			X		Display
3278-3	X			X	3270-X	Display
3278-4	X			X	3270-X	Display
3278-5	X			X	3270-X	Display
3279-2A	X			X		Display
3279-2B	X			X	3270-X	Display
3279-2C	X			X		Display
3279-3A	X			X		Display
3279-3B	X			X	3270-X	Display
3279-2X	X			X	3270-X	Display
3279-3X	X			X	3270-X	Display
3279-S2B	X			X	3270-X	Display
3279-S3G	X			X	3270-X	Display
3290	X			X	3270-X	Display
2740	X					Communication terminal

Maximum and Default Specifications for AREA and SEG

Unit	Full Capability AREA Size		Status Display AREA Size		Screen Size	SEG Default
	Max	Default	Max	Default		
3270-X	*	*	*	*	N/A	*
3277-2	19	14	23	11,12	24 x 80	9
3278-2	20	14	24	12,12	24 x 80	10
3278-2A	16	14	20	10,10	20 x 80	8
3278-3	28	14	32	16,16	32 x 80	14
3278-4	39	14	43	21,22	43 x 80	19
3278-5	24	14	27	13,14	27 x 132	12
3279-2A	20	14	24	12,12	24 x 80	10
3279-2B	20	14	24	12,12	24 x 80	10
3279-2C	16	14	20	10,10	20 x 80	8
3279-2X	20	14	24	12,12	24 x 80	10
3279-3A	28	14	32	16,16	32 x 80	14
3279-3B	28	14	32	16,16	32 x 80	14
3279-3X	28	14	32	16,16	32 x 80	14
3279-S2B	20	14	24	12,12	24 x 80	10
3279-S3G	28	14	32	16,16	32 x 80	14
3290	20	14	24	12,12	24 x 80	10
3290	24	14	27	13,14	27 x 132	12
3290	27	14	31	15,16	31 x 80	13
3290	28	14	31	15,16	31 x 160	14
3290	28	14	32	16,16	32 x 80	14
3290	39	14	43	21,22	43 x 80	19
3290	46	14	50	25,25	50 x 106	23
3290	58	14	62	31,31	62 x 80	29
3290	59	14	62	31,31	62 x 160	28

Member Name: CSVLLAxx

Use of the Member

Installations use the CSVLLAxx member to specify which libraries library lookaside (LLA) is to manage and how it is to manage them. The START LLA,LLA=xx command identifies the CSVLLAxx parmlib member to be used to build the LLA directory. The START LLA,LLA=xx command is usually issued by the IEACMD00 parmlib member during system initialization; the command can also be entered by the system operator. See *System Commands* for the complete syntax of the START LLA command.

Installations also use CSVLLAxx members to specify data sets to be added or removed from LLA, to change the way LLA is managing a data set, and/or to specify the entry point names or the LLA libraries to be selectively refreshed in the LLA directory. For this use, the operator specifies the CSVLLAxx parmlib member in a MODIFY LLA,UPDATE=xx command. Selectively modifying the LLA directory allows updated LLA-managed modules to be activated without activating other changed LLA-managed modules. Selective LLA refresh also avoids the purging and restaging of modules that have not changed. When a staged module is refreshed in the LLA directory, LLA purges the copy of the module in the virtual lookaside facility (VLF) data space and may then restage the module into VLF.

If you wish to use a parmlib data set other than SYS1.PARMLIB for LLA, specify the data set LLA is to use on an //IEFPARM DD statement in LLA's start JCL procedure.

Use the PARMLIB and SUFFIX keywords in the CSVLLAxx member to point to another CSVLLAxx member in a parmlib data set. The CSVLLAxx member pointed to by the PARMLIB and SUFFIX keywords must not point back to the original member.

Parameter in IEASYSxx (or issued by the operator):

None

Syntax Rules

The syntax for CSVLLAxx is:

- Data, including comments, must be contained in columns 1-71; the system ignores columns 72-80.
- There is no limit to the number of times a keyword can be specified.
- Commas or blanks in any combination constitute a delimiter.
- Delimiters or comments can precede the keywords.
- Delimiters are not required between keywords with value; the right parenthesis after the specified keyword is sufficient.
- Comments must begin with "/*" and end with "*/" Comments are allowed between keywords or values.
- The LLA command will not be processed if there are any errors in the contents of the CSVLLAxx parmlib member(s).

Syntax Format

```
LIBRARIES{(libname1,libname2,libname3,...)  }
         {   [MEMBERS(mmbr1,mmbr2,...)]     }
         {(-LNKLST-)                        }
         {   [MEMBERS(mmbr1,mmbr2,...)]     }

LNKMEMBERS(epn1,epn2,epn3,...)

REMOVE(dsnx,dsny,...{-LNKLST-},...)

PARMLIB(dsn) SUFFIX(xx)

NOFREEZE(libname1,libname2,...{-LNKLST-},...)

FREEZE(libname1,libname2,...{-LNKLST-},...)
```

IBM-Supplied Defaults

None.

Internal Parameters

Keyword	Options	Meaning and Use	Default Value
LIBRARIES	(*libname1,libname2,libname3...*)	The LIBRARIES statement lists the names of libraries (*libname1,libname2,libname3*) that are to be added to LLA, if they are not already being managed by LLA, or selectively refreshed, if they are already being managed by LLA.	None
		A library name (*libname1,libname2,libname3,*) must be 1 to 44 characters long.	
		-LNKLST- (the hyphens are coded) can be used to designate the whole LNKLST concatenation. It is a shorthand way to identify all the data sets in the LNKLST instead of listing each of them, that is: (*-LNKLST-,production libname1,production libname2*)	None
MEMBERS	(mmbr1,mmbr2,...)	The MEMBERS statement provides the programmer a method to refresh dynamically the specified members in the named production libraries.	None
		The MEMBERS statement must be preceded by a LIBRARIES statement that identifies the libraries contains the new versions of the members.	
		The member names (*mmbr1,mmbr2...*) must be 1 to 8 characters long.	
		The LLA directory for each of the listed libraries will be updated with the directory entries for the listed members found in each of the data sets. That is, LIBRARIES(LIB.1, LIB.2, LIB.3) MEMBERS(A,B,C) will result in the LLA directory for LIB.1 being refreshed with directory entries found for members A, B and C in the DASD directory for LIB.1; the LLA directory for LIB.2 being refreshed with directory entries found for members A, B and C in the DASD directory for LIB.2, and so forth.	
		If *-LNKLST-* is in the LIBRARIES statement list, LLA will dynamically refresh the specified members in the LNKLST. Only the first occurrence of the member name in the LNKLST concatenation will be used to refresh the LLA directory.	
LNKMEMBERS	(*epn1,epn2,epn3...*)	The LNKMEMBERS statement lists the entry point names (*epn1,epn2,epn3...*) that are in the LNKLST concatenation and that are to be selectively refreshed. LNKMEMBERS is equivalent to specifying *LIBRARIES(-LNKLST-) MEMBERS(epn1,epn2,epn3...)*	None
		The entry point names (*epn1,epn2,epn3...*) must be 1 to 8 characters long.	
REMOVE	(*libname1,libname2,...*)	The REMOVE(*libname1,libname2,...*) statement allows you to remove dynamically libraries from the list of libraries managed by LLA.	None
		LLA will no longer be used to provide directory entries or staged modules for these libraries.	
		The REMOVE(*-LNKLST-,...*) statement allows you to remove dynamically each of the LNKLST production libraries from the libraries managed by LLA. LLA will no longer be used to provide directory entries or staged modules for the LNKLST libraries. This statement does not affect the LNKLST concatenation itself. You cannot change the LNKLST concatenation once the system is IPLed.	

Keyword	Options	Meaning and Use	Default Value
PARMLIB	(*dsn*)		
SUFFIX	(*xx*)	The PARMLIB (*dsn*) SUFFIX(*xx*) statement allows you to specify another parmlib member that contains more LLA control statements to be processed. The parmlib member is processed completely when this control statement is encountered. (*dsn*) identifies the data set where the the CSVLLAxx member identified by (*xx*) should be found. The keywords allow you to include LLA control statements from data sets other than SYS1.PARMLIB, thereby allowing system programmers to control LLA's specifications without being given update access to SYS1.PARMLIB.	None
NOFREEZE | FREEZE	(*libname1,libname2,...*)	The NOFREEZE | FREEZE statement allows the system programmer to specify for a library whether to have LLA use the directory it maintains in its own storage (FREEZE) or to use the directory on auxiliary storage (NOFREEZE). *-LNKLST-* can be used to designate the whole LNKLST concatenation. It is a shorthand way to identify all the data sets in the LNKLST instead of listing each of them, that is, (*-LNKLST-,non-LNKLST libname1,non-LNKLST libname2*) When a LLA library is specified with FREEZE, the installation takes full advantage of LLA's I/O reduction for directory search for the library and for load module fetch. When a LLA library is specified with NOFREEZE, the installation does not get the benefit of reduced I/O while searching the library's directory but it can still get the increased performance benefit of fetching the load modules from VLF instead DASD. An installation can change the FREEZE or NOFREEZE status of a LLA library at any time by using the MODIFY LLA command. When an LLA library is specified with FREEZE, the user of that library always gets the LLA version of a member; thus if the user makes an update to the member and then tries to browse it, the user sees the LLA version of the member. The LLA version does not have the changes in it until LLA is refreshed. If a user does multiple linkedits to a member in a FREEZE data set, the base for each subsequent linkedit does not include the previous linkedits; the base is the LLA version of the member. The recommended way to make updates in the production system is to use IEBCOPY under FREEZE mode. The member to be updated should be copied to another data set, the linkedits run against the second data set and then the updated member can be copied back to the LLA-managed data set. If LLA-managed production libraries must be updated directly, LLA should be refreshed to managed the data set in NOFREEZE mode.	NOFREEZE FREEZE for LNKLST data sets accessed through the LNKLST concatenation.

Member Name: EXSPATxx

Use of the Member

The EXSPATxx member allows an installation to specify actions to be taken to recover from excessive spin conditions without operator involvement. A system routine might be spinning because it cannot obtain a resource held by another processor. EXSPATxx allows you to specify the action or actions to be taken to end the spin loop if the excessive spin is detected while spinning for one of the following:

- A RISGNL response
- A spin lock needs to be released
- A successful BIND BREAK
- The RESTART resource
- An address space to quiesce
- An INTSECT release condition

EXSPATxx does not support actions if the spin loop is due to a SIGP failure.

In a normal environment, the IBM-supplied defaults for the system are sufficient. In a test environment, however, you might want to specify particular actions to be taken.

In addition to recovery actions, EXSPATxx also allows you to specify a timeout interval. If an excessive spin continues beyond the timeout interval, the system automatically issues a SPIN action before taking any recovery action specified in an EXSPATxx member. If the SPIN action does not end the excessive spin, the excessive spin continues until the timeout interval is reached. Then the recovery actions specified in the EXSPATxx member are taken.

If you do not specify an EXSPATxx member of SYS1.PARMLIB, the system issues the following automatic spin loop recovery actions after 10 seconds:

```
SPIN,ABEND,TERM,ACR
```

Note: The default spin loop timeout interval is 10 seconds in an MVS environment. For MVS running on VM or in a non-dedicated PR/SM environment, the default spin loop timeout interval is 40 seconds.

See *Recovery and Reconfiguration* for more information on handling excessive spin conditions.

Parameter in IEASYSxx (or issued by the operator):

None

Syntax Rules

The following syntax rules apply to EXSPATxx:

- SPINRCVY or SPINTIME= can start in any column between column 1 and 71.
- SPINRCVY or SPINTIME= must be the first entry on a record.
- Comments must begin with /* and end with */.
- Any number of blanks can exist between SPINRCVY and the first action.
- Multiple actions on a SPINRCVY statement must be separated by a comma. A blank is not a valid delimiter between two actions.

- SPINRCVY and multiple actions must appear on the same logical record.
- SPINTIME=*sss* cannot contain blanks.
- The system ignores columns 71 through 80.

A Syntax Example

An example of the syntax for EXSPATxx is:

```
SPINRCVY   action[,action]...
SPINTIME=sss
```

IBM-Supplied Defaults

None.

The installation must add EXSPATxx to SYS1.PARMLIB. You can issue the SET EXS=xx command to change the parmlib member after initialization. Two alphanumeric characters are appended to EXSPAT to form the name of the EXSPATxx parmlib member.

If you do not create a EXSPATxx member, the IBM-supplied default recovery actions for excessive spin conditions are:

```
SPIN,ABEND,TERM,ACR
SPINTIME=10
```

Note: The default spin loop timeout interval is 10 seconds in an MVS environment. For MVS running on VM or in a non-dedicated PR/SM environment, the default spin loop timeout interval is 40 seconds.

Example of EXSPATxx

If you specify more than one action, the system executes the requests in the order of their specification. For example, if you code

```
SPINRCVY      ABEND,ACR
SPINTIME=30
```

The system allows the excessive spin to continue for about 30 seconds then issues a SPIN action before your first recovery action. If the excessive spin condition is not alleviated, the system issues an informational message and allows the excessive spin to continue for another 30 seconds. The system then issues the ABEND to end the unit of work that is causing the spin. The recovery routines try to recover; if they fail, the excessive spin continues for 30 seconds. Finally, the system issues the ACR for the alternate CPU recovery.

Internal Parameters

Statement	Parameter	Meaning and Use	Default Value
SPINRCVY		The SPINRCVY statement allows you to specify the actions the system is to take to end a spin loop not caused by a SIGP failure. SPINRCVY must be the first entry on a record defining these actions. At least one action is required on a SPINRCVY statement. You can specify a maximum of eight parameters on a SPINRCVY statement.	
	ABEND	The system issues an ABEND to the current unit of work. Recovery routines will try to recover.	
	ACR	The system invokes alternate CPU recovery (ACR) processing for the processor causing the excessive spin.	
	OPER	The system issues message IEA331A and processes the operator's reply. If the message could not be issued, the processor that is in an excessive spin is put into a restartable wait state (X'09n'). The operator may respond to the wait state with ABEND, ACR, TERM, SPIN, or U to continue.	
	SPIN	Specifies the system is to return to the caller and to continue spinning. An informational message will be displayed.	
	TERM	The system issues an ABEND to the current unit of work. Recovery routines will **not** try to recover.	
SPINTIME=*sss*		SPINTIME allows you to specify the excessive spin loop timeout interval where *sss* is the number of seconds. *sss* can be one through three digits but must be higher than the IBM-supplied default. In a native MVS environment, the default is 10 seconds. For MVS running on VM or in a non-dedicated PR/SM environment, the default spin loop timeout interval is 40 seconds.	10 seconds 40 seconds See the explanation.

Member Name: GRSCNFxx

Use of the Member

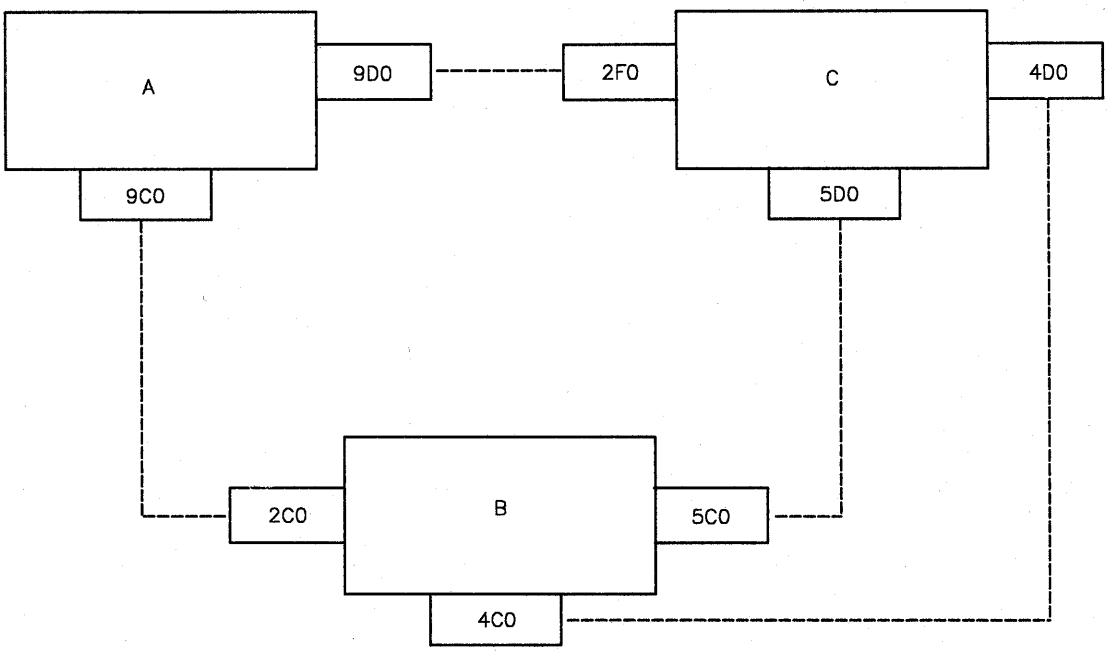
When your installation decides to use global resource serialization to serialize access to global resources (usually data sets on shared DASD volumes) among units of work on multiple systems, the contents of the GRSCNFxx parmlib member are used during system initialization to define the global resource serialization complex. A *global resource serialization complex* consists of two or more systems connected by communication links. The systems in the complex use global resource serialization to control access to data sets on shared DASD volumes at the data set level rather than at the volume level. (See *Planning: Global Resource Serialization* for more information.)

To define a global resource serialization complex, use the GRSCNFxx parmlib member. GRSCNFxx contains a GRSDEF statement for each system. The statement identifies:

- The name of the system in the complex, by means of the MATCHSYS internal parameter.
- The CTC links assigned to the system, by means of the CTC internal parameter.
- The amount of time that the RSA-message is to spend in the system, by means of the RESMIL internal parameter.
- Whether or not the system identified by the MATCHSYS keyword has the automatic restart capability, by means of the RESTART internal parameter.
- Whether or not the system identified by the MATCHSYS keyword has the capability to automatically rejoin the complex, by means of the REJOIN internal parameter.
- The amount of time, the tolerance limit, before GRS will detect a time-out, by means of the TOLINT internal parameter.
- The method to establish the GRS ring acceleration function, by means of the ACCELSYS internal parameter.

See "Internal Parameters" later in this description for complete syntax information about these internal parameters.

Figure 3-4 shows a design for a global resource serialization complex consisting of three systems (A, B, and C) and the device numbers of the CTC links that connect the systems. The figure also shows the contents of a GRSCNFxx member that defines the complex.




```
GRSCNFxx
GRSDEF          /* GRSDEF FOR SYSTEM A                    */
  MATCHSYS (A)
  CTC(9C0)      /* 9C0 IS THE CTC DEVICE NUMBER           */
  CTC(9D0)      /* 9D0 IS THE CTC DEVICE NUMBER           */
  RESMIL(30)    /* RESIDENCY TIME IS 30 MILLISECONDS      */
  RESTART(NO)   /* SYSTEM A DOES NOT HAVE THE
                   AUTOMATIC RESTART CAPABILITY           */
GRSDEF          /* GRSDEF FOR SYSTEM B                    */
  MATCHSYS (B)
  CTC(2C0)      /* 2C0 IS THE CTC DEVICE NUMBER           */
  CTC(4C0)      /* 4C0 IS THE CTC DEVICE NUMBER           */
  CTC(5C0)      /* 5C0 IS THE CTC DEVICE NUMBER           */
  RESMIL(30)    /* RESIDENCY TIME IS 30 MILLISECONDS      */
  RESTART(YES)  /* SYSTEM B DOES HAVE THE
                   AUTOMATIC RESTART CAPABILITY           */
GRSDEF          /* GRSDEF FOR SYSTEM C                    */
  MATCHSYS (C)
  RESMIL(30)    /* RESIDENCY TIME IS 30 MILLISECONDS      */
  CTC(2F0)      /* 2F0 IS THE CTC DEVICE NUMBER           */
  CTC(5D0)      /* 5D0 IS THE CTC DEVICE NUMBER           */
  CTC(4D0)      /* 4D0 IS THE CTC DEVICE NUMBER           */
  RESTART(YES)  /* SYSTEM C DOES HAVE THE
                   AUTOMATIC RESTART CAPABILITY           */
```

Figure 3-4. Definition of a Global Resource Serialization Complex

There are two basic ways to use GRSCNFxx to define your complex:

1. You can create one GRSCNFxx parmlib member that contains GRSDEF statements that define all of the systems in the complex. After creating the member, copy it to SYS1.PARMLIB on each system. As each system is initialized, it reads the GRSCNFxx member, locates its own GRSDEF statement, and uses the information in the statement during initialization.
2. You can create a unique GRSCNFxx member for each system in the complex. The member consists of a GRSDEF statement for that particular system. You place the unique member in SYS1.PARMLIB on that particular system, and that system uses the information it contains during initialization.

Variations or combinations of the two methods are, of course, possible.

Whichever method you choose, GRSCNFxx provides information about the configuration of the complex that global resource serialization requires during initialization. Other information that global resource serialization requires comes from the following system parameters:

- GRS=option, which indicates whether this system is to start a global resource serialization complex (GRS=START), join an existing complex (GRS=JOIN), or not be part of a global resource serialization complex (GRS=NONE).
- GRSCNF=xx, which identifies the GRSCNFxx parmlib member to be used.
- GRSRNL=xx or GRSRNL=(xx,yy...), which identifies the GRSRNLxx parmlib member(s) to be used.
- SYSNAME=name, which identifies the name of the system in the global resource serialization complex (and matches the name specified for the system in a GRSDEF statement in GRSCNFxx).

The IBM-supplied default is GRS=JOIN. To avoid initialization and processing overhead, override the default for a system that is not to be part of a global resource serialization complex. For the system that is to start the complex, you can either override the default or instruct the operator to respond with START to message ISG009D. Once the global resource serialization complex is initialized, there is no operational distinction between the system that started the complex and any system that joined the complex.

The GRSCNF=xx parameter identifies the GRSCNFxx parmlib member to be used to initialize the system, where xx corresponds to the last two characters in the name of the GRSCNFxx member to be used. GRSCNF=xx is meaningful only when a system is initialized with GRS=START or GRS=JOIN; it is ignored when a system is initialized with GRS=NONE.

The GRSRNL parameter specifies one or more GRSRNLxx parmlib members, which contain resource name lists (RNLs). Global resource serialization uses the RNLs to determine how to treat the resources that are named in the RNLs. The GRSRNL parameter is meaningful only when a system is initialized with GRS=START or GRS=JOIN; it is ignored when a system is initialized with GRS=NONE.

The SYSNAME=name parameter specifies the name the system is to be known by in the global resource serialization complex. For consistency and simplicity, you might find it helpful to use the same name as you use in the SID SMF parameter. Global resource serialization initialization routines use this name as a search

argument to find the correct GRSDEF statement in the specified GRSCNFxx parmlib member. When the matching GRSDEF statement is found, the information in the statement is used to initialize the system.

The GRS, GRSCNF, GRSRNL, and SYSNAME parameters can be specified only at IPL time, either in IEASYSxx or by the operator. They all remain in effect for the duration of the IPL. For specific details on how to specify any one of these parameters, see the description of the parameter provided later in Part 3 under IEASYSxx.

During initialization processing for a system that is to join an existing complex, global resource serialization verifies that the information in GRSCNFxx is consistent with the existing complex. Even if a system is defined in a GRSCNFxx parmlib member and initialized with GRS=JOIN, it cannot join the complex unless at least one CTC link associated with it in GRSCNFxx is attached to another active system in the complex.

Parameters in IEASYSxx:

GRS= { **JOIN** | **START** | **NONE** }

GRSCNF=xx

GRSRNL= { **xx** | **(xx,yy...)** }

SYSNAME=name

Syntax Rules

GRSCNFxx can contain GRSDEF statements and comments. The following rules apply to the creation of GRSCNFxx by means of the IEBUPDTE utility:

- Use columns 1 through 71; columns 72 through 80 are ignored.
- Comments begin with "/*" and end with "*/." A comment can span card images and can appear anywhere except within a keyword or a specified value.
- Each GRSDEF statement is defined as beginning with the characters "GRSDEF" and ending with the character immediately preceding the next GRSDEF statement.

- The GRSDEF statement format is:

```
GRSDEF [ MATCHSYS  { (name) } ]
       [           { (*)    } ]
       [RESMIL (1-8 decimal digit number)]
       [CTC (CTC link device number)...]
       [ RESTART  { (YES) } ]
       [          { (NO)  } ]
       [ REJOIN   { (YES) } ]
       [          { (NO)  } ]
       [TOLINT (1-8 decimal digit number)]
       [ACCELSYS (1-2 decimal digit number)]
```

Notes:

1. You can put a blank anywhere except within a keyword or a specified value.
2. You can use as many card images as you need for one GRSDEF statement.
3. You can put multiple parameters on one card image.
4. No delimiters or blanks are needed between parameters. For example, the following GRSDEF statement is valid:

   GRSDEF MATCHSYS(*) CTC(9A0)CTC(8B0)CTC(7D0)RESMIL(30)

5. The value for the MATCHSYS name must consist of from 1 to 8 alphanumeric characters, including the @,#, and $ characters. Should you want to, you can specify "NONAME," which is the default for the SYSNAME system parameter.
6. Duplicate GRSDEF statements cause a syntax error. A syntax error also occurs if you specify more than one GRSDEF statement with MATCHSYS(*), either explicitly or by default, or more than one GRSDEF statement with the same MATCHSYS name.
7. Each GRSDEF statement must include at least one CTC parameter and can include up to a maximum of 64 CTC parameters.

IBM-Supplied Default

There is no default GRSCNFxx parmlib member. However, there are default parameter values, as follows:

MATCHSYS(*)RESMIL(30)RESTART(YES)ACCELSYS(99)REJOIN(YES)

Internal Parameters

Parameter	Meaning and Use
MATCHSYS { (name) / (*) }	Identifies the system defined in the GRSDEF statement. A system being initialized compares the value specified on the SYSNAME system parameter to the value specified for MATCHSYS to locate its GRSDEF statement in GRSCNFxx. Omitting MATCHSYS or specifying MATCHSYS(*) is regarded as a match unless a GRSDEF statement exists with an explicit name that matches the SYSNAME value. Including both a GRSDEF statement with an explicit MATCHSYS name and a GRSDEF statement with MATCHSYS(*), either explicitly or by default, thus does not cause a syntax error; the system ignores the GRSDEF statement with MATCHSYS(*). **Note:** To form a ring of two or more systems, the MATCHSYS name for each system must be unique. Global resource serialization initialization processing searches the statements in GRSCNFxx until it finds a GRSDEF statement that matches the SYSNAME value, then uses the information in the statement to continue initialization. It issues an error message if it finds no match.
RESMIL(number)	Specifies, in milliseconds, the RSA-message residency time (that is, the amount of time that the RSA-message is to spend in this system). If you omit RESMIL, then the default for the RSA-message residency time is 30 milliseconds. Careful selection of the RESMIL value ensures that the RSA-message moves adequately from system to system within the ring and that each system in the ring has enough time to process its workload after passing on the RSA-message.
CTC(device number)	Identifies the device number of a CTC link attached to this system that is dedicated to the use of global resource serialization. The device number must be a 3-digit hexadecimal address; leading and trailing zeroes must be specified. Only one device number can be specified on each CTC parameter, but up to 64 different CTC parameters can be specified.
RESTART { (YES) / (NO) }	Indicates whether the system defined in the MATCHSYS parameter can automatically rebuild a disrupted ring. A disrupted ring occurs when the RSA-message stops moving because of a system failure or a link failure.
REJOIN { (YES) / (NO) }	Indicates whether the system defined in the MATCHSYS parameter can automatically rejoin the active ring, after that system was stopped, then started. This would eliminate the need for the operator to issue a VARY GRS, RESTART command to have the system join the active ring.
TOLINT {(NUM)}	Specifies in seconds, the tolerance time interval, (GVTOLINT), GRS adds to the expected time it expects the RSA-message to return to that system, before it considers that system overdue. The TOLINT value specified must be greater than zero (0), and less than 86,000 seconds (24 hours). When the TOLINT value specified is < = 0 or > = 86,000 seconds than the message ISG008E is issued, the TOLINT keyword is ignored, the default values for GVTOLINT and GVTMEINT are used, and the IPL continues. When the TOLINT value is specified correctly GRS will also use new algorithms which eliminate the need to specify the event interval (GVTMEINT).
ACCELSYS {(num)}	Specifies the threshold for GRS ring acceleration and the number of systems that must see a resource request before it is granted. To use ring acceleration, every system must have a link to every other system; ring acceleration requires a fully connected complex, and it also requires an alternate link for each connection. When the ACCELSYS keyword is not specified ring acceleration is turned off. The ACCELSYS threshold value range is from two (2) through ninety nine (99). The maximum performance benefit is provided when the ACCELSYS threshold value is 2. The ACCELSYS default value is ninety nine (99);

Member Name: GRSRNLxx

Use of the Member

GRSRNLxx consists of three resource name lists (RNLs). When a global resource serialization complex is active, the system uses these RNLs to determine how to treat the resources defined in the RNLs. The RNLs are:

- The **SYSTEM inclusion RNL**. The system treats each resource named in this RNL as a global resource if an ENQ or a DEQ macro instruction for the resource specifies a scope of SYSTEM.
- The **SYSTEMS exclusion RNL**. The system treats each resource named in this RNL as a local resource if an ENQ or a DEQ macro instruction for the resource specifies a scope of SYSTEMS.
- The **RESERVE conversion RNL**. The system suppresses a hardware reserve for each resource named in this RNL if a RESERVE macro instruction requests the use of the resource.

See *Planning: Global Resource Serialization* for detailed information on global resource serialization and RNLs.

Parameter in IEASYSxx (or issued by the operator):

GRSRNL = { **aa** | **(aa,bb...)** }

The two alphameric characters, represented by aa (or bb, etc.), are appended to GRSRNL to form the name of the GRSRNLxx member(s).

Syntax Rules

GRSRNLxx can contain RNLDEF statements and comments. The following rules apply to the creation of GRSRNLxx using the IEBUPDTE utility:

- Use columns 1 through 71; columns 72 through 80 are ignored. Note that if you need to continue to another card image when specifying a value (such as the name on the RNAME parameter), you must start the continuation in column 1.
- Comments begin with "/*" and end with "*/". A comment can span card images and can appear anywhere except within a keyword or a specified value.
- Each RNLDEF statement is defined as beginning with the characters "RNLDEF" and ending with the character immediately preceding the next RNLDEF statement.
- Each RNLDEF statement must contain, in any order, the RNL, TYPE, QNAME, and RNAME parameters. For example:

```
RNLDEF RNL(INCL) TYPE(SPECIFIC) QNAME(SYSDTNM)
       RNAME(SYS1.USR)
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSDTNM)
       RNAME(SYS1.U)
RNLDEF RNAME(SYS1.U) TYPE(GENERIC) RNL(INCL)
       QNAME(SYSDTNM)
```

Exceptions to this are:

- The RNAME parameter may be omitted if TYPE(GENERIC) is specified for a generic QNAME resource. For example:

  ```
  RNLDEF TYPE(GENERIC) QNAME(X'1A4A783F2B') RNL(EXCL)
  ```

- If the RNLs are to be obtained from SYS1.LINKLIB, then LINKLIB(YES) must be specified on an RNLDEF statement, either by itself or with the other valid parameters. For example:

  ```
  RNLDEF LINKLIB(YES)
  ```

  or

  ```
  RNLDEF TYPE(GENERIC) QNAME(SYSDSN) RNAME(SYS1.U30LIB)
         LINKLIB(YES)
  ```

  Note: If you specify LINKLIB(YES) with other valid parameters, these parameters are ignored. However, invalid parameters that precede LINKLIB(YES) will receive a syntax error; invalid parameters that follow LINKLIB(YES) are ignored.

- A null GRSRNLxx parmlib member is valid. The system does not consider a member in error if it is empty; that is, if it contains neither RNLDEF statements nor comments.

Notes:

1. You can put a blank anywhere except within a keyword or a specified value.
2. You can use as many card images as you need for one RNLDEF statement.
3. You can put multiple parameters on one card image.
4. You do not need to put a blank (or other delimiter) between parameters. For example, the following RNLDEF statement is valid:

   ```
   RNLDEF RNL(EXCL)TYPE(SPECIFIC)QNAME(SYSDSN)RNAME(SYS1.DAE)
   ```

5. You can use from 1 to 8 characters for the name that you specify on the QNAME parameter. You must enclose the name in parentheses.
6. You can use from 1 to 255 characters for the name that you specify on the RNAME parameter. You must enclose the name in parentheses. Any complete line of blanks (columns 1-71) will be ignored when specifying a value for the RNAME.
7. You can specify the names on the QNAME and RNAME parameters in any of the following formats:

   - If the name contains nondisplayable characters, you must use two hexadecimal digits to specify each character of the name. For example:

     ```
     RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(X'18')
            RNAME(X'19')
     ```

   - If the name contains displayable characters that are alphameric (A-Z and 0-9), #, @, and $, and/or a period (.), you enter the name as is. For example:

     ```
     RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(STW@7)
            RNAME(REW.20)
     ```

- If the name contains displayable characters other than those already described (including a blank, but excluding a single quotation mark), you must enclose the name in single quotation marks. For example:

```
RNLDEF RNL(CON) TYPE(SPECIFIC) QNAME('$ ( )*')
       RNAME('A B')
```

IBM-Supplied Default

IBM provides the following statements in GRSRNL00 (the default member):

```
/*****************************************************************/
/*   RNLDEF STATEMENT SPECIFYING LINKLIB(YES). IF THIS STATEMENT  */
/*   EXISTS, RNLS WILL BE LOADED FROM THE ISGGRNL0 MEMBER OF      */
/*   SYS1.LINKLIB. IN ORDER TO HAVE RNLS LOADED FROM  THE         */
/*   GRSRNL00 MEMBER OF SYS1.PARMLIB, REMOVE THIS STATEMENT.      */
/*****************************************************************/
RNLDEF LINKLIB(YES)
/*****************************************************************/
/*   SYSTEMS EXCLUSION RESOURCE NAME LIST - RNLDEF STATEMENTS     */
/*****************************************************************/
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(PASSWORD)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(SYS1.BRODCAST)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(SYS1.DAE)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(SYS1.DCMLIB)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSDSN) RNAME(SYS1.DUMP)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(SYS1.LOGREC)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSDSN) RNAME(SYS1.MAN)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(SYS1.NUCLEUS)
RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(SYSDSN) RNAME(SYS1.PAGE)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(SYS1.STGINDEX)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(SYS1.SVCLIB)
RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSDSN) RNAME(SYS1.UADS)
/*****************************************************************/
/*    SYSTEM INCLUSION RESOURCE NAME LIST - RNLDEF STATEMENTS    */
/*****************************************************************/
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSDSN)
/*****************************************************************/
/*    RESERVE CONVERSION RESOURCE NAME LIST - RNLDEF STATEMENTS   */
/*  (THE DEFAULT RESERVE CONVERSION RESOURCE NAME LIST IS EMPTY)  */
/*****************************************************************/
```

Internal Parameters

Parameter	Meaning and Use
LINKLIB { (YES) / (NO) }	Specifies the source of the RNLs, where YES indicates the RNLs are to be loaded from SYS1.LINKLIB and NO indicates the RNLs are to be obtained from the GRSRNLxx member(s). The default is LINKLIB(NO).
RNL { (INCL) / (EXCL) / (CON) }	Specifies the RNL in which the resource name entry is to be placed, where INCL indicates the SYSTEM inclusion RNL, EXCL indicates the SYSTEMS exclusion RNL, and CON indicates the RESERVE conversion RNL.
TYPE { (GENERIC) / (SPECIFIC) }	Specifies the type of resource name entry being defined in the RNL. For a SPECIFIC entry, you must specify a QNAME (major name) and an RNAME (minor name) for the resource.
QNAME(name)	Specifies the major name of the resource.
RNAME(name)	Specifies the minor name of the resource.

Member Name: GTFPARM (or an installation-supplied name)

Use of the Member

GTFPARM provides default or installation-defined trace options to control the generalized trace facility (GTF). The member is read only when the operator (or an automatic command) issues START GTF. It is not used during system initialization.

The procname of the START command can name the IBM-supplied cataloged procedure, GTF. The PROC statement of that procedure identifies GTFPARM as the member from which GTF will get its trace parameters. If the installation wants to substitute another member in place of GTFPARM, the operator may enter the replacement member name on the START command with the MEMBER keyword.

The IBM procedure, GTF, as supplied in SYS1.PROCLIB, contains these statements:

```
//GTF PROC          MEMBER=GTFPARM
//IEFPROC EXEC      PGM=AHLGTF,PARM='MODE=EXT,DEBUG=NO,
                    TIME=NO',TIME=1440,REGION=2880K
//IEFRDER DD        DSNAME=SYS1.TRACE,UNIT=SYSDA,
//                  SPACE=(4096,20),DISP=(NEW,KEEP)
//SYSLIB DD         DSN=SYS1.PARMLIB(&MEMBER),DISP=SHR
```

For further analysis of this procedure and additional information about GTF trace options, see the GTF topics in *Dump and Trace Services.*

Because default options in GTFPARM specify minimal data for only a limited number of traced events, you may wish to expand GTF capabilities through one of the following methods:

- Specify another SYS1.PARMLIB member name, using the MEMBER keyword on the START command.
- Change the trace options in GTFPARM, using the IEBUPDTE utility.
- Change the SYSLIB DD statement of the IBM procedure to specify a parmlib member, which you create via IEBUPDTE, that has the options you want.
- Retain the IBM procedure to handle default options, and write one or more alternate procedures, each specifying a different alternate parmlib member. You could design each member to contain GTF options useful under particular circumstances. Instruct the operator when to issue the START command for each procname.

GTF tries to read parameters from the specified parmlib member. If an error occurs in opening or reading the member, or if GTF detects a syntax error, it writes a diagnostic message to the operator, and requests him to SPECIFY TRACE OPTIONS, as if no GTF parmlib member were available. The operator therefore must have a complete list of desired GTF parameters available when he starts GTF.

Parameter in IEASYSxx (or issued by the operator):

None

Syntax Rules

The following rules apply to the creation of a GTF parmlib member by means of the IEBUPDTE utility:

- Specify the TRACE keyword and its main options only on the first record. Do not place them on subsequent records. For example,

  ```
  Record #1:      TRACE=IOP,SVCP,SSCH
  ```

  This example requests the tracing of specific I/O interrupts, specific SVC interrupts, and all start subchannel and resume subchannel operations.

- The second and subsequent records should contain only "prompting" keywords, such as IO= or SVC=. These keywords provide for detailed operands that indicate which I/O interrupts or which SVC interrupts should be traced. For example, the IOP and SVCP keywords in the Record #1 example (above) must be followed by prompting records that name specific device numbers and specific SVC numbers for which interrupts should be traced. As an example,

  ```
  Record #2:      IO=(191,192,193),SVC=(1,2,3)
  ```

  If the specific operands of any prompting keyword are missing, GTF does not prompt the operator. It accepts a general specification. For example, if IOP is specified in Record #1, and Record #2 specifies only SVC=(1,2,3), and no particular device numbers are specified for I/O tracing, GTF assumes that tracing of I/O interrupts is desired for *all* devices.

  When all of the records have been read, GTF issues message AHL103I TRACE OPTIONS SELECTED -- IO,SVC=(1,2,3). The operator can then respond to message AHL125A RESPECIFY TRACE OPTIONS OR REPLY U by either entering all the desired options or accepting the input that was specified in the parmlib member.

- An END keyword or an end-of-file must follow all prompting keywords. If the END keyword is used, it must appear either on the last record supplying prompting keywords or on its own record.

- If you need to specify additional operands for the same keyword, restate the keyword and the additional operands in a subsequent prompting record. The previous examples, expanded to include additional SVC numbers and an END keyword, would appear like this:

  ```
  Record #1:      TRACE=IOP,SVCP,SSCH
  Record #2:      IO=(191,192,193),SVC=(1,2,3)
  Record #3:      SVC=(4,5,6,7,8,9,10),END
  ```

- Certain trace options do not work in combination with others. Figure 3-5 shows those trace options that should not be specified together. If you specify two or more options from the same horizontal row, GTF uses the option in the lowest numbered column and ignores the other options. For example, if you specify both SYSP and SVC (row C), GTF uses SYSP and ignores SVC.

	1	2	3	4	5
A	SYSM	SYSP	SYS	SSCHP	SSCH
B	SYSM	SYSP	SYS	IOP	IO
C	SYSM	SYSP	SYS	SVCP	SVC
D	SYSM	SYSP	SYS	PIP	PI
E	SYSM	SYSP	SYS	EXT	
F	SYSM	SYSP	SYS	RR	
G	SYSM	SYSP	SYS	CSCH	
H	SYSM	SYSP	SYS	HSCH	
I	SYSM	SYSP	SYS	MSCH	
J	USRP	USR			
K	CCWP	CCW			

Figure 3-5. Combining Certain GTFPARM Options

IBM-Supplied Defaults

When GTF is started by specifying the IBM-supplied cataloged procedure, the following options exist in GTFPARM:

```
TRACE=SYSM,USR,TRC,DSP,PCI,SRM
```

These keywords cause the following events to be recorded: SVC interruptions, I/O interruptions, program-controlled I/O (PCI) interruptions, program interruptions, external interruptions, dispatcher executions, start subchannel and resume subchannel operations, clear subchannel operations, halt subchannel operations, modify subchannel operations, entries to the system resource manager, entries to recovery routines, events associated with GTF (TRC), and data passed to GTF via the GTRACE macro (USR). All keywords except USR result in minimal format trace entries. USR entries are the length specified by the user in the GTRACE macro. Such entries may optionally be a maximum of 256 bytes, excluding the prefix.

Note: If you use the default options provided, be aware that some data will be duplicated by the system trace if it is executing concurrently with GTF.

Internal Parameters

Parameter	Meaning and Use
ASIDP	This parameter identifies one to five address spaces for which GTF is to trace events. The option is only valid if specified with a GTF option that causes event tracing, such as IO or SVC. If you specify ASIDP, GTF prompts for the address space identifiers and traces only those events occurring in the identified address spaces. **Note:** If you specify the JOBNAMEP option with ASIDP, GTF traces events for (1) jobs identified by JOBNAMEP, regardless of the address spaces in which they run, and (2) address spaces identified by ASIDP, regardless of the jobs running in the address spaces. Therefore, you could get event tracing for address spaces other than those identified by ASIDP.
CCW	This parameter requests recording of channel programs and their associated data. The option is only valid if specified with the IO and/or SSCH options. The CCW and IO options produce a trace of channel programs for I/O interruptions on those devices for which I/O interruptions are to be traced. The CCW and SSCH options produce a trace of channel programs for start subchannel and resume subchannel operations on those devices for which SSCH events are to be traced.
CCWP	This parameter is similar to CCW, except that GTF prompts for CCW keyword values, such as the number of bytes of data to be traced for each CCW. (CCWP allows you to override the defaults set for the CCW keywords.) The option is only valid if specified with the IO and/or SSCH options.
CSCH	This parameter requests recording for all clear subchannel operations.
DSP	This parameter requests recording for all dispatchable units of work (SRB, LSR, TCB, and SVC prologue dispatch events). The option is not included in the specification of SYS or SYSM. It must be specified in addition to other parameters. The parameter produces comprehensive format except when SYSM is also specified. With SYSM, data is in minimal format.
END	This parameter indicates the end of the prompting records. In the parmlib member, an end of file serves the same purpose. Never include the END parameter in the first record (the record that contains the TRACE parameters) because GTF regards such an occurrence as an error.
EXT	This parameter requests comprehensive recording for all external interruptions.
HSCH	This parameter requests recording for all halt subchannel operations.
IO	This parameter requests recording of all non-program controlled I/O interruptions. To obtain recording of program-controlled I/O interruptions specify PCI. Both options may be specified.
IOP	This parameter requests the same type of recording as does IO, except that GTF prompts for the numbers of specific devices whose I/O interruptions are to be recorded.
JOBNAMEP	This parameter identifies one to five jobnames for which GTF is to trace events. The option is only valid if specified with a GTF option that causes event tracing, such as IO or SVC. If you specify JOBNAMEP, GTF prompts for the jobnames and traces only those events occurring when the identified jobs execute. **Note:** If you specify the ASIDP option with JOBNAMEP, GTF traces events for (1) address spaces identified by ASIDP, regardless of the jobs running in the address spaces, and (2) jobs identified by JOBNAMEP, regardless of the address spaces in which they run. Therefore, you could get event tracing for jobs other than those identified by JOBNAMEP.
MSCH	This parameter requests recording for all modify subchannel operations.
PCI	This parameter requests that program-controlled I/O interruptions (PCIs) be recorded in the same format as other requested I/O trace records. If specific device numbers are specified through prompting records, program controlled I/O interruptions are recorded for the specified devices. I/O tracing must be requested because PCI must be specified with a GTF parameter that causes IO, such as SYS or SYSM.
PI	This parameter requests comprehensive recording for all program interruptions (0-255).
PIP	This parameter is similar to PI except that GTF prompts the operator for the specific interruption codes for which data is to be recorded.
RNIO	This parameter requests recording of all VTAM network activity. If you specify both SYSM and RNIO, GTF records minimal trace data for RNIO. Otherwise, GTF records comprehensive trace data for RNIO.
RR	This parameter requests that uses of recovery routines (FRRs and STAE/ESTAE routines) be recorded. The trace record is created when the recovery routine returns control to the recovery termination manager (RTM). Data is in comprehensive format except when the option is specified via SYSM.

Parameter	Meaning and Use
SLIP	This parameter requests a trace entry either each time a match occurs for a SLIP trap that specifies a tracing action or each time the system encounters the SLIP trap with the SLIP DEBUG option specified. The SLIP command specifies the amount of data and the type of SLIP trace the system is to build. The SLIP option is not included in the specification of the SYS or SYSM option and must be specified additionally. Specifying the SYS or SYSM option does not affect the data collected on the SLIP trace record.
SRM	This parameter requests a trace entry each time that the system resource manager (SRM) is invoked. The option is not included in the specification of SYS or SYSM. It must be specified in addition to other parameters. Data is in comprehensive format except when SYSM is also specified. Comprehensive format includes the jobname.
SSCH	This parameter requests recording for all start subchannel and resume subchannel operations.
SSCHP	This parameter is similar to SSCH, except that it requests GTF to prompt for the specific device numbers (such as 151, 152) for which SSCH events should be recorded. Only the devices that the operator or the parmlib member specifies cause SSCH entries in the trace.
SVC	This parameter requests comprehensive recording of all SVC interruptions.
SVCP	This parameter is similar to SVC, except that GTF prompts for specific SVC numbers for which data is to be recorded.
SYS	This parameter requests comprehensive trace data for all external interruptions (EXT), program interruptions (PI), recovery routines (RR), and supervisor call interruptions (SVC). SYS causes recording of all I/O interruptions (IO), start subchannel and resume subchannel operations (SSCH), halt subchannel operations (HSCH), and modify subchannel operations (MSCH). When you specify DSP, RNIO, or SRM, in addition to SYS, GTF produces comprehensive trace data for those events. **Note:** Specification of SYS, SYSM, or SYSP causes GTF to ignore the following trace options if you specify them in any form: CSCH, HSCH, MSCH, SSCH, IO, SVC, PI, EXT, RR.
SYSM	This parameter requests recording of minimal trace data for all external interruptions (EXT), program interruptions (PI), recovery routines (RR), and supervisor call interruptions (SVC). SYSM causes recording of all I/O interruption (IO), start subchannel and resume subchannel operations (SSCH), clear subchannel operations (CSCH), halt subchannel operations (HSCH), and modify subchannel operations (MSCH). When you specify DSP, RNIO, or SRM, in addition to SYSM, GTF produces minimal trace data for those events. **Note:** Specification of SYS, SYSM, or SYSP causes GTF to ignore the following trace options if you specify them in any form: CSCH, HSCH, MSCH, SSCH, IO, SVC, PI, EXT, RR.
SYSP	This parameter requests comprehensive recording for the same events as the SYS option, but causes GTF to prompt for specific SVC, IO, SSCH, and PI events that you want recorded. When you specify DSP, RNIO, or SRM, in addition to SYSP, GTF produces comprehensive trace data for those events. **Note:** Specification of SYS, SYSM, or SYSP causes GTF to ignore the following trace options if you specify them in any form: CSCH, HSCH, MSCH, SSCH, IO, SVC, PI, EXT, RR.
TRC	This parameter requests that traced events include those related to GTF processing itself. If this parameter is not specified, GTF-related events are excluded from the trace output.
USR	This parameter requests that user data passed to GTF via the GTRACE macro be recorded with the system data.
USRP	This parameter is similar to USR, except that GTF prompts for user event numbers for which data is to be recorded.

Member Name: IEAABD00

Use of the Member

IEAABD00 contains IBM defaults and/or installation assigned parameters for ABDUMP, for use when an ABEND dump is written to a SYSABEND data set.

ABDUMP parameters for a dump to be taken to a SYSABEND data set may be specified as follows:

- The dump request parameter list pointed to by the DUMPOPT keyword of an ABEND macro instruction.[3] The list can be built by using the list form of the SNAP macro. (See *Supervisor Services and Macro Instructions* for details regarding the ABEND and SNAP macros.)
- The initial system dump options specified in IEAABD00. These options are added to the options on the dump request parameter list.
- The system dump options as altered by the CHNGDUMP command. With the CHNGDUMP command, options can be added to or deleted from the system dump options list. The CHNGDUMP command can also cause the dump request parameters to be ignored. (For a description of the CHNGDUMP command refer to *System Commands*.)

The ABDUMP initialization routine reads IEAABD00 to get ABDUMP parameters. If during initialization, IEAABD00 is found to be invalid or can't be located, the operator is notified. No prompting occurs. If both valid and invalid options are included in the member, or a syntax error is encountered, a message lists the valid options that were accepted before the error occurred.

Parameter in IEASYSxx (or specified by the operator):

None

[3] An ABEND dump can also be requested by a CALLRTM or SETRP macro. See *SPL: System Macros and Facilities*.

Syntax Rules

The following rules apply to the replacement of IEAABD00 by means of the IEBUPDTE utility:

- There are two keywords, SDATA and PDATA. Each keyword is followed by a string of operands separated by commas and enclosed in parentheses. A single operand does not need parentheses.

  Examples:

  ```
  SDATA=(SQA,CB,ENQ,TRT)            or  SDATA=ALLSDATA
  PDATA=(PSW,REGS,SA,ALLPA,SPLS)    or  PDATA=ALLPDATA
  ```

- Normally both parameters (i.e., SDATA = operands and PDATA = operands) can fit on one card image. If, however, continuation is needed, use a comma followed by a blank.

  Example:

  ```
  SDATA=(SQA,CB,ENQ,TRT),
  PDATA=(PSW,REGS,
  SA,ALLPA,SPLS)
  ```

- To include a comment, place the comment on the same line as the data. Use at least one blank to separate the data from the comment. The following example shows a continuation and the related comments.

  Example:

  ```
  SDATA=(SQA,CB,  SQA and CB data
  ENQ,TRT)        ENQ and trace data
  ```

IBM-Supplied Defaults

The following defaults are placed in IEAABD00 by IBM:

```
SDATA=(LSQA,CB,ENQ,TRT,ERR,DM,IO,SUM),
PDATA=(PSW,REGS,SPLS,ALLPA,SA)
```

These options request a dump of the following areas:

- LSQA, including subpools 229 and 230
- formatted control blocks for the task
- formatted global resource serialization control blocks for the task
- GTF trace and/or system trace
- recovery termination control blocks for the task
- data management control blocks for the task
- IOS control blocks for the task
- summary data (see explanation under SUM parameter)
- PSW at entry to ABEND
- contents of general registers at entry to ABEND
- save area linkage information and a backward trace of save areas
- modules listed on the link pack area queue for the task and the job pack area queue for the task, and active SVC modules related to the failing task
- user storage allocated for the task

Internal Parameters

Parameter		Meaning
SDATA=		
	ALLSDATA	Except ALLVNUC and NOSYM, all the following options are automatically specified.
	The following parameters request dump of specific SDATA areas, as indicated:	
	ALLVNUC	Entire virtual nucleus. SQA, LSQA, and the PSA are included.
	NOSYM	No symptom dump is to be produced.
	SUM	Requests that the dump contain summary data, which includes the following: • Dump title. • Abend code and PSW at the time of the error. • If the PSW at the time of the error points to an active load module: (1) the name and address of the load module, (2) the offset into the load module indicating where the error occurred, and (3) the contents of the load module. • Control blocks related to the failing task. • Recovery termination control blocks. • Save areas. • Registers at the time of the error. • Storage summary consisting of 1K (1024) bytes of storage before and 1K bytes of storage after the addresses pointed to by the registers and the PSW. The storage will be printed only if the user is authorized to obtain it, and, when printed, duplicate addresses will be removed. • System trace table entries for the dumped address space.
	NUC	Read/write portion of the control program nucleus. SQA, LSQA and the PSA are included.
	PCDATA	Program call information for the task being dumped.
	SQA	The system queue area.
	LSQA	Local system queue area for the address space. If storage is allocated for subpools 229 and 230, they will be dumped for the current task.
	SWA	Scheduler work area used for the failing task.
	CB	Control blocks related to the failing task.
	ENQ	Global resource serialization control blocks for the task.
	TRT	System trace table and GTF trace, as available.
	DM	Data management control blocks (DEB, DCB, IOB) for the task.
	IO	IOS control blocks (UCB, EXCPD) for the task.
	ERR	Recovery termination control blocks (RTM2WA, registers from the SDWA, SCB, EED) for the task.
PDATA=		
	ALLPDATA	All the following options are automatically specified.
	The following parameters request dump of specific PDATA areas, as indicated:	
	SUBTASKS	Problem data (PDATA) options requested for the designated task will also be in effect for its subtasks.
	PSW	Program status word at entry at ABEND.
	REGS	Contents of general registers at entry to ABEND.
	SA or SAH	SA requests save area linkage information and a backward trace of save areas. This option is automatically selected if ALLPDATA is specified. SAH requests only save area linkage information.
	JPA	Contents of the job pack area (module names and contents) that relate to the failing task.
	LPA	Contents of the LPA (module names and contents) related to the failing task. Includes active SVCs related to the failing task.
	ALLPA	Contents of both the job pack area and the LPA, as they relate to the failing task, plus SVCs related to the failing task.
	SPLS	User storage subpools (0-127) related to the failing task.

Member Name: IEAAPFxx

Use of the Member

IEAAPFxx is a list of program library names (dsnames) and corresponding volume serial numbers that require APF authorization. (APF means the authorized program facility.) SYS1.LINKLIB and SYS1.SVCLIB are automatically authorized. SYS1.LPALIB, however, is not automatically authorized (except during NIP) because it is closed at the end of NIP processing and is not required until the next IPL when the PLPA is to be reloaded.

If an installation wants IEAAPFxx, it must explicitly create the member via the IEBUPDTE utility. The member IEAAPF00 must be explicitly created by the installation also.

Notes:

1. Except for concatenations opened during NIP, any unauthorized library that is concatenated to authorized libraries will cause all of the concatenated libraries to be considered unauthorized.
2. You can specify a maximum of 253 library names in an IEAAPFxx member.

Parameter in IEASYSxx (or specified by the operator):

APF=xx

The two-character identifier xx is appended to IEAAPF to identify the IEAAPFxx member. If the APF-parameter is not specified, only SYS1.LINKLIB and SYS1.SVCLIB will be authorized (i.e., placed in the APF table).

Syntax Rules

The following rules apply to the creation of IEAAPFxx by means of the IEBUPDTE utility:

- Place only one library name and corresponding volume serial number on a record (card image).
- Duplicate data set names are valid.
- On each record, first enter the library name, then one or more blanks, then the volume serial number.
- To continue to another record, place a comma after the volume serial number. Omit this comma on the last record.

  Example:

```
first record:     LIB087      614703,
second record:    LIB122      705650
```

IBM-Supplied Default

If neither IEAAPFxx nor IEAAPF00 exists, SYS1.SVCLIB and SYS1.LINKLIB are authorized. Additionally, if the default for the LNKAUTH system parameter is taken (LNKAUTH=LNKLST) or is specified in IEASYSxx or by the operator, libraries in the LNKLST concatenation are also APF authorized when accessed as part of the LNKLST concatenation.

Internal Parameters

Not applicable

Member Name: IEAAPP00

Use of the Member

IEAAPP00 contains the names of authorized installation-written I/O appendage routines. These appendages, when listed, can be used by any unauthorized user program. Otherwise, only programs authorized under APF or running under system protection key (0-7) may use the EXCP appendages. If your installation does not use EXCP appendages, you need not create IEAAPP00.

IEAAPP00 can be built during the sysgen process, if the installation specifies any of the appendage keywords of the DATASET macro. The possible keywords and the associated appendage types are:

SIOAPP	Start I/O appendages
CHEAPP	channel end appendages
EOEAPP	end-of-extent appendages
PCIAPP	PCI appendages
ABEAPP	abnormal end appendages

NIP accesses SYS1.PARMLIB, reads the list of appendage names in IEAAPP00, builds an appendage name table, and sets a pointer to the table in the CVT. On each subsequent OPEN, the appendage name table will be examined, instead of IEAAPP00. If the EXCP caller is not in system protection key or the job step is not authorized, Open verifies that the caller's appendage names are listed. If the names can't be found, open issues a 913 ABEND. If, however, the caller is authorized, open loads the appendages without inspecting the list.

IEAAPP00 can be created at sysgen, with all appendage names specified, even though all appendages have not yet been created. The IEBCOPY step of sysgen will, in this case, issue a diagnostic message, but will not fail the step. The installation thus has the ability to "prime" IEAAPP00 with appendage names whose modules can be created later. To create or alter IEAAPP00 after sysgen, use the IEBUPDTE utility. Then re-IPL.

Rules for Specifying Appendages with the DATASET Macro at SYSGEN

- Use the appendage keywords (e.g., SIOAPP=) for user appendages that are to be used by unauthorized problem programs. The full name of an appendage is eight characters long. The first six characters must be IGG019. The last two characters, specified as the operands of the appendage keywords, can range from WA to Z9. Example: CHEAPP=XZ,ZZ,Z6.
- Specify LPALIB or SVCLIB as the system library parameter.
- Do not use the MEMBERS=keyword (unless the appendages are used exclusively by authorized programs). The MEMBERS keyword would prevent sysgen from building IEAAPP00.
- Each appendage keyword can list up to 84 appendage-name suffixes (e.g., XZ).
- Omit a particular appendage keyword if there are no appendages of that type. For example, do not specify EOEAPP if end-of-extent appendages are not desired.
- Omit all appendage keywords, if there are to be no user-written appendages.

Syntax Example for DATASET Macro:

The following example shows how the sysgen DATASET macro can be used to specify appendage names:

```
DATASET     LPALIB,
            PDS=MYLIB, (user data set from which the
                        appendage modules are to be
                        copied to LPALIB)
            MEMBERS=(IGG019XX,IGG019WA),
            SIOAPP=(XY),
            CHEAPP=(XZ,ZZ,ZY)
```

In this example six appendages are copied to LPALIB. The two appendages, IGG019XX and IGG019WA, are copied to LPALIB but are not listed in IEAAPP00. These two are to be used only by authorized programs. The other four appendages, a Start I/O and three channel-end appendages, are copied to LPALIB and are listed in IEAAPP00.

Syntax Rules

The following rules apply to the construction of IEAAPP00 by means of the IEBUPDTE utility:

- IEAAPP00 can contain up to five entries, each entry containing names of a particular type of appendage (e.g., abnormal-end or Start I/O). You need not necessarily use all five types of entries.
- Each entry consists of an appendage-type name, followed by a list of suffixes of that type, separated by commas. The appendage-type name can start in any column. For example:

  ```
  SIOAPP        WA,W1
  ```

  This entry specifies two Start I/O appendages.
- Indicate continuation, as with most parmlib members, by means of a comma followed by at least one blank. The next record can start in any column.

Syntax Example:

Here is an example of a complete IEAAPP00 member:

```
SIOAPP        Y1,Y2,
EOEAPP        X1,W2,X3,X4,X5,X6,
PCIAPP        X3
```

In this example, note that there are no channel-end appendages and none for abnormal end. Routine IGG019X3 is used as both end-of extent and PCI appendage.

IBM-Supplied Default

If IEAAPP00 doesn't exist, only IGG019E4 (channel end/abnormal end appendage for interactive terminal facility) is placed in the table of authorized appendage routines.

Internal Parameters

Not applicable

Member Name: IEACMD00

Use of the Member

IEACMD00 contains IBM-supplied commands, as follows:

- SLIP commands to suppress dumps that are considered unneeded because the system provides information (such as a message) that is normally sufficient for problem determination. For selected abend codes, the SLIP commands in IEACMD00 suppress either all dumps or specific types of dumps.

  You might prefer copying the SLIP commands from IEACMD00 to an IEASLPxx parmlib member. By using the IEASLPxx member, you avoid IEACMD00 restrictions. For example, IEASLPxx supports multiple-line commands and IEACMD00 does not. IEASLPxx does not require any special command syntax; IEACMD00 does.

- A CHNGDUMP command to add trace table and LSQA information to SVC dumps.
- A SET command (SET DAE=00) to indicate that the system is to use the ADYSET00 parmlib member to start DAE processing.
- A START command (START LLA, SUB=MSTR) to start the Library lookaside (LLA) procedure, which resides in SYS1.PROCLIB. The LLA procedure then starts the Library lookaside function.
- A START command (START BLSJPRMI, SUB=MSTR) creates IPCS tables which allows SNAP ABDUMP and IPCS to print formatted control blocks in dumps.

An installation that uses the optional COMMNDxx parmlib member should determine if there are commands in COMMNDxx that conflict with commands in IEACMD00, and resolve any conflicts. It is also recommended that an installation place its commands in the COMMNDxx member, leaving the IEACMD00 member for IBM-supplied commands only. Place all SLIP commands in IEASLPxx.

Notes:

1. An installation that does not currently use SLIP commands should realize that the execution of IEACMD00 causes the system to allocate fixed storage for the SLIP processing modules and for the control blocks that define the SLIP traps supplied by IBM in IEACMD00. Approximately 30K bytes of storage will be fixed in the extended LPA for the SLIP processing modules and approximately 1K bytes of storage will be fixed in the extended SQA for the control blocks needed by the SLIP traps.
2. The order in which task creating commands, such as SLIP, appear in IEACMD00 does NOT guarantee the order in which they are executed. Thus, IEACMD00 should not be used for commands that must be executed in a specific order. Commands are issued in the order that they appear in COMMNDxx, but they are executed as follows:
   - Immediate commands, such as DISPLAY T, are executed sequentially as they are issued from COMMNDxx.

- Execution of task-creating commands, such as DISPLAY A, are deferred until system initialization is complete. Then, factors such as multitasking, multiprocessing, and competition for resources influence the order in which these commands are executed.

  Thus, COMMNDxx should *not* be used to issue task-creating commands that must be executed in a specific order, because the execution order of these commands can vary.

3. For an SVC dump suppressed by a SLIP command with the action of NODUMP or NOSVCD, a SYS1.LOGREC entry will contain a code indicating the reason for the suppression. For an ABEND dump suppressed by a SLIP command with the action of NODUMP, NOSYSU, NOSYSA, or NOSYSM, message IEA848I will indicate the reason for the suppression.
4. If a program loop causes an "out-of-space" abnormal termination (as in abend code B37) and you want a dump, specify a SYSABEND DD statement or a SYSMDUMP DD statement.
5. The SET DAE=00 command causes the ADYSET00 parmlib member to execute. As a result, dump analysis and elimination (DAE) processing as specified in ADYSET00 is in effect.

   If your installation does not want to activate DAE automatically, modify ADYSET00 to contain DAE=STOP.

   For more information on DAE and its parmlib members, see the description of ADYSETxx earlier in "Part 3: System Initialization."
6. See *System Commands* for descriptions of the CHNGDUMP, SET, START, and SLIP commands.

Parameter in IEASYSxx (or issued by the operator):

None

Syntax Rules

The following rules apply to the modification of IEACMD00 by means of the IEBUPDTE utility:

- Enter only one command for each card image.
- Enter the COM= keyword, followed by the command enclosed in apostrophes.
- Do not specify continuation on any card image.

IBM-Supplied Defaults

The following commands are placed in IEACMD00 by IBM:

```
COM='CHNGDUMP SET,SDUMP=(LSQA,TRT),ADD'
COM='SLIP SET,C=013,ID=X013,A=NOSVCD,J=JES2,END'
COM='SLIP SET,C=028,ID=X028,A=NOSVCD,END'
COM='SLIP SET,C=0E7,ID=X0E7,A=NOSVCD,END'
COM='SLIP SET,C=0F3,ID=X0F3,A=NODUMP,END'
COM='SLIP SET,C=13E,ID=X13E,A=NODUMP,END'
COM='SLIP SET,C=222,ID=X222,A=NODUMP,END'
COM='SLIP SET,C=322,ID=X322,A=NODUMP,END'
COM='SLIP SET,C=33E,ID=X33E,A=NODUMP,END'
COM='SLIP SET,C=622,ID=X622,A=NODUMP,END'
COM='SLIP SET,C=804,ID=X804,A=(NOSVCD,NOSYSU),END'
COM='SLIP SET,C=806,ID=X806,A=(NOSVCD,NOSYSU),END'
COM='SLIP SET,C=80A,ID=X80A,A=(NOSVCD,NOSYSU),END'
COM='SLIP SET,C=9FB,ID=X9FB,A=NOSVCD,J=JES3,END'
COM='SLIP SET,C=B37,ID=XB37,A=(NOSVCD,NOSYSU),END'
COM='SLIP SET,C=D37,ID=XD37,A=(NOSVCD,NOSYSU),END'
COM='SLIP SET,C=E37,ID=XE37,A=(NOSVCD,NOSYSU),END'
COM='SET DAE=00'
COM='START LLA,SUB=MSTR'
COM='START BLSJPRMI,SUB=MSTR'
```

Note: If you use a name other than JES2 or JES3 to start the job entry subsystem, the SLIP commands for abend codes X'013' and X'9FB' have no effect. Therefore, to suppress SVC dumps for these abend codes, if you are not using JES2 or JES3, place your own SLIP command in COMMNDxx. Specify NOSVCD on the ACTION = keyword (A =) and the name of your job entry subsystem on the JOBNAME = keyword (J =).

Internal Parameters

Not applicable

Member Name: IEADMP00

Use of the Member

IEADMP00 contains IBM defaults and/or installation parameters for ABDUMP for use when an ABEND dump is written to a SYSUDUMP data set.

ABDUMP parameters for a SYSUDUMP data set may be specified as follows:

- The dump request parameter list pointed to by the DUMPOPT keyword of an ABEND macro instruction.[4] The list can be built by using the list form of the SNAP macro. (See ABEND and SNAP macros in *Supervisor Services and Macro Instructions* for details.)
- The initial system dump option specified in IEADMP00. This option is added to the options on the dump request parameter list.
- The system dump options as altered by the CHNGDUMP command. With the CHNGDUMP command, options can be added to or deleted from the system dump options list. The CHNGDUMP command can also cause the dump request parameters to be ignored. (For a description of the CHNGDUMP command refer to *System Commands.*

During IPL an information message will notify the operator if IEADMP00 is invalid or can't be found. No prompting of the operator will occur. If the member contains both valid and invalid parameters, an information message will indicate the valid options that were accepted before the error occurred.

Parameter in IEASYSxx (or specified by the operator):

None

---

[4] An ABEND dump can also be invoked by the CALLRTM and SETRP macros. (For details, see *SPL: System Macros and Facilities.*)

Syntax Rules

The following rules apply to the replacement of IEADMP00 by means of the IEBUPDTE utility:

- There are two keywords, SDATA and PDATA. Each keyword is followed by a string of operands separated by commas and enclosed in parentheses. A single operand does not need parentheses.

  Examples:

  ```
  SDATA=(SQA,CB,ENQ,TRT)            or    SDATA=ALLSDATA

  PDATA=(PSW,REGS,SA,ALLPA,SPLS)    or    PDATA=ALLPDATA
  ```

- Normally both parameters (SDATA = operands and PDATA = operands) can fit on one card image. If, however, continuation is needed, use a comma followed by a blank.

  Example:

  ```
  SDATA=(SQA,CB,ENQ,TRT),
  PDATA=(PSW,REGS,
  SA,ALLPA,SPLS)
  ```

- To include a comment, place the comment on the same line as the data. Use at least one blank to separate the data from the comment. The following example shows a continuation and related comments.

  Example:

  ```
  SDATA=(SQA,CB,  SQA and CB data
  ENQ,TRT)        ENQ and trace data
  ```

IBM-Supplied Defaults

The following default is automatically placed in IEADMP00 by sysgen.

```
SDATA=SUM
```

See the SUM parameter (under the following topic "Internal Parameters") for the data provided in a dump when the default is in effect.

Internal Parameters

Parameter		Meaning and Use
SDATA=		
	ALLSDATA	Except ALLVNUC and NOSYM, all the following options are automatically specified.
	The following parameters request dump of specific SDATA areas, as indicated:	
	ALLVNUC	Entire virtual nucleus. SQA, LSQA, and the PSA are included.
	NOSYM	No symptom dump is to be produced.
	SUM	Requests that the dump contain summary data, which includes the following: • Dump title. • Abend code and PSW at the time of the error. • If the PSW at the time of the error points to an active load module: (1) the name and address of the load module, (2) the offset into the load module indicating where the error occurred, and (3) the contents of the load module. • Control blocks related to the failing task. • Recovery termination control blocks. • Save areas. • Registers at the time of the error. • Storage summary consisting of 1K (1024) bytes of storage before and 1K bytes of storage after the addresses pointed to by the registers and the PSW. The storage will be printed only if the user is authorized to obtain it, and, when printed, duplicate addresses will be removed. • System trace table entries for the dumped address space.
	NUC	Read/write portion of the control program nucleus. SQA, LSQA, and the PSA are included.
	PCDATA	Program call information for the task being dumped.
	SQA	The system queue area.
	LSQA	Local system queue area for the address space. If storage is allocated for subpools 229 and 230, they will be dumped for the current task.
	SWA	Scheduler work area used for the failing task.
	CB	Control blocks related to the failing task.
	ENQ	Global resource serialization control blocks for the task.
	TRT	System trace table and GTF trace, as available.
	DM	Data management control blocks (DEB, DCB, IOB) for the task.
	IO	IOS control blocks (UCB, EXCPD) for the task.
	ERR	Recovery termination control blocks (RTM2WA, registers from the SDWA, SCB, EED) for the task.
PDATA=		
	ALLPDATA	All the following options are automatically specified.
	The following parameters request dump of specific PDATA areas, as indicated:	
	PSW	Program status word at entry to ABEND.
	REGS	Contents of general registers at entry to ABEND.
	SA or SAH	SA requests save area linkage information and a backward trace of save areas. This option is automatically selected if ALLPDATA is specified. SAH requests only save area linkage information.
	JPA	Contents of the job pack area that relate to the failing task. These include module names and contents.
	LPA	Contents of the LPA related to the failing task. These include module names and contents. Also includes active SVCs related to the failing task.
	ALLPA	Contents of both the job pack area and the LPA, as they relate to the failing task, plus SVCs related to the failing task.
	SPLS	User storage subpools (0-127) related to the failing task.
	SUBTASKS	Problem data (PDATA) options requested for the designated task will also be in effect for its subtasks.

Member Name: IEADMR00

Use of the Member

IEADMR00 contains IBM defaults and/or installation parameters for ABDUMP, for use when an ABEND dump is written to a SYSMDUMP data set.

ABDUMP parameters for a SYSMDUMP data set may be specified as follows:

- The dump request parameter list pointed to by the DUMPOPT keyword of an ABEND macro instruction.[5] The list can be built by using the list form of the SNAP macro. (See ABEND and SNAP macros in *Supervisor Services and Macro Instructions* for details.)
- The initial system dump options specified in IEADMR00. These options are added to the options on the dump request parameter list.
- The system dump options as altered by the CHNGDUMP command. With the CHNGDUMP command, options can be added to or deleted from the system dump options list. The CHNGDUMP command can also cause the dump request parameters to be ignored. (For a description of the CHNGDUMP command, see *System Commands*

During IPL, an information message will notify the operator if IEADMR00 is invalid or can't be found. No prompting of the operator will occur. If the member contains both valid and invalid parameters, an information message will indicate the valid options that were accepted before the error occurred.

Parameter in IEASYSxx (or specified by the operator):

None

[5] An ABEND dump can also be invoked by the CALLRTM and SETRP macros. (For details, see *SPL: System Macros and Facilities*.)

Syntax Rules

The following rule applies to the replacement of IEADMR00 by means of the IEBUPDTE utility:

- There is one keyword, SDATA. The keyword is followed by a string of operands separated by commas and enclosed in parentheses. A single operand does not need parentheses.

  Example:

  ```
  SDATA=(SQA,TRT)          or    SDATA=ALLSDATA
  ```

- To include a comment, place the comment on the same line as the data. Use at least one blank to separate the data from the comment. The following example shows a continuation and the related comments.

  Example:

  ```
  SDATA=(SQA,LSQA,  SQA and LSQA data
  SWA,TRT)          SWA and trace data
  ```

IBM-Supplied Defaults

The following defaults are automatically placed in IEADMR00 by sysgen.

```
SDATA=(NUC,SQA,LSQA,SWA,TRT,RGN,SUM)
```

These options request a dump of the following areas:

- read/write portion of the control program nucleus
- system queue area
- local system queue area
- scheduler work area
- GTF trace and/or system trace
- private area for the address space region
- summary data (see explanation under SUM parameter)

Internal Parameters

Parameter		Meaning and Use
SDATA=		
	ALLSDATA	Except ALLNUC and NOSYM, all the following options are automatically specified.
	The following parameters request dump of specific SDATA areas, as indicated:	
	NUC	Read/write portion of the control program nucleus. The PSA is included.
	SQA	The system queue area.
	LSQA	Local system queue area for the address space. If storage is allocated for subpools 229 and 230, they will also be included.
	SWA	Scheduler work area used for the failing task.
	TRT	System trace table and GTF trace, as available.
	LPA	Contents of the LPA related to the failing task. These include module names and contents. Also includes SVCs related to the failing task.
	CSA	Common service area subpools that are not fetch protected.
	RGN	Private area for the address space region.
	GRSQ	All global resource serialization control blocks and other global resource serialization storage.
	ALLNUC	Entire control program nucleus. The PSA is included.
	NOSYM	No symptom dump is to be produced.
	SUM	Summary dump information as provided in the SVC dump. (See the description of the SUM parameter for the SDUMP macro instruction in *SPL: System Macros and Facilities*.)

Member Name: IEAFIXxx (Fixed LPA List)

Use of the Member

IEAFIXxx contains the names of modules from SYS1.SVCLIB, the LPALST concatenation, and the LNKLST concatenation that are temporarily fixed in real storage for the duration of an IPL. The member may be used to temporarily add or replace SVC or ERP routines that already exist in the pageable LPA, or to page fix such routines to improve system performance. Like the temporary modules chosen through the MLPA option, fixed LPA modules are not automatically reactivated by a quick start or a warm start IPL. That is, the fixed LPA can be reestablished only by re-specification of the FIX parameter at the quick start or warm start IPL.

Because fixed modules are not paged, you can save I/O time and paging overhead by placing moderately used modules from the LPALST concatenation into the fixed LPA. When a module is requested, the program manager searches the list of fixed routines before it examines the pageable LPA directory. The price for this performance improvement is the reduction in real storage available for paging old jobs and starting new jobs. Remember that pages referenced frequently tend to remain in real storage even when they are not fixed.

You can use the fixed LPA to reduce page-fault overhead at the expense of some real storage. This trade-off is desirable with a system that tends to be CPU bound but that has sufficient real storage. You implement the trade-off by placing moderately used modules from the LPALST concatenation into the fixed LPA (via parmlib member IEAFIXxx). High usage PLPA modules probably need not be fixed because they are referenced frequently enough to remain in real storage anyway. Because less real storage will be available for pageable programs, the system resource manager will swap out address spaces that would otherwise occupy real storage, awaiting CPU availability.

Modules specified in IEAFIXxx are loaded and fixed in the order in which they are named in the member. The modules' CDEs are placed on the active LPA queue for easy access by the program manager. (**Note:** To keep search time within reasonable limits, do not allow the fixed link pack area to become excessively large.) If the first load module of a type 3 or 4 SVC routine is specified, the SVC table is updated as required.

Modules specified in IEAFIXxx are placed in the FLPA or in the extended FLPA, depending on the RMODE of the modules. Modules with an RMODE of 24 are placed in the FLPA, while those with an RMODE of ANY are placed in the extended FLPA. By default, the FLPA and the extended FLPA are page protected, which means that a protection exception will occur if there is an attempt to store data into either area. To override page protection, use the NOPROT option on the FIX system parameter.

Note: Any modules specified for either fixed LPA list or MLPA must come from authorized libraries. There may be non-authorized libraries present in the LNKLST concatenation.

Fixed LPA modules may optionally be specified at sysgen by means of the RESIDNT keyword of the DATASET macro. Be careful not to specify the same module name in both the RESIDNT and the MEMBERS keywords, because the two keywords are mutually exclusive. These modules are listed in member IEAFIX01. To specify this list at IPL, the IEASYSxx member or the operator should include FIX=01 as a system parameter.

Note: The FLPA and the extended FLPA are not mapped V=R. Modules in either area cannot be assumed to be backed by contiguous real frames.

Parameter in IEASYSxx (or specified by the operator):

FIX= { **aa** | **(aa[,L][,NOPROT])** | **(aa,bb,...[,L][,NOPROT])** }

The two alphameric characters aa, bb, etc. are appended to IEAFIX to form the name of one or more IEAFIXxx members of SYS1.PARMLIB. NIP defaults to no fixed LPA, if you fail to specify the option in one of the following ways:

- FIX keyword included in the IEASYSxx member.
- FIX keyword entered by the operator at IPL.

The LPA modules that are fixed in storage are also page protected, by default. If an attempt is made to store into a page-protected module, a protection exception occurs. However, an installation can use the NOPROT option to override the page protection default. When NOPROT is specified, the LPA modules in the IEAFIXxx parmlib member(s) are not page protected in storage.

If the L option is specified, the system displays the contents of the IEAFIXxx parmlib member(s) at the operator's console as the system processes the member(s).

Syntax Rules

The following rules apply to the creation of IEAFIXxx by means of the IEBUPDTE utility:

- On the first record, start with the data set name -- SYS1.SVCLIB, SYS1.LPALIB (meaning the LPALST concatenation), or SYS1.LINKLIB (meaning the LNKLST concatenation) -- from which modules will be taken. Follow the name with at least one blank.
- After the blank, list the modules to be loaded, separated by commas. Do not put a blank before or after a comma that separates module names on a record. You can use all columns except 72 through 80.
- Indicate continuation of the list by a comma after the last name on all records, except the last.
- Do not include the data set name on continued records.
- Place a *new* data set name at the beginning of a record, followed by at least one blank and a new string of module names.
- You can use major names or alias names, or both. Any alias names by which a module is to be accessed must be included. If an alias name is included and the associated major name is omitted, the system will locate the major name.

- End the last record with one or more blanks.

Syntax Example:

```
1st record     SYS1.LINKLIB      IKJPARS,IKJPARS2,IKJSCAN,
                                 IKJEFD00,IKJDAIR,
2nd record     SYS1.SVCLIB       IGC0009C,
3rd record     IGC09301,IGC09302,IGC09303
```

IBM-Supplied Defaults

None

Internal Parameters

This category is not applicable because module names, not parameters, appear in this member.

Member Name: IEAICSxx

Use of the Member

IEAICSxx contains the installation control specification (ICS) that associates units of work (transactions) with performance groups. Performance groups are used by the system resources manager to control and report on transactions. Specifying an IEAICSxx parmlib member is optional. For more information, see "Installation Control Specification Concepts" in "Part 5: The System Resources Manager."

Parameter in IEASYSxx (or specified by the operator):

$$\text{ICS} = \left\{ \begin{array}{l} \text{xx} \\ \text{(xx[,L])} \\ \text{(,L)} \end{array} \right\}$$

The two-character identifier, xx, is appended to IEAICS to specify one of several possible parmlib members from which the system resources manager obtains its parameters. If the L option is specified, the system displays the contents of the IEAICSxx parmlib member at the operator's console as the system processes the member. (For more information, see "ICS=xx" in the description of the IEASYSxx member.)

Syntax Rules

See "Part 5: The System Resources Manager."

IBM-Supplied Defaults

None

Member Name: IEAIPSxx

Use of the Member

Used by the system resources manager to define performance characteristics that are assigned to transactions. For more information see "IPS Concepts" in "Part 5: The System Resources Manager."

Parameter in IEASYSxx (or specified by the operator):

$$\text{IPS} = \left\{ \begin{array}{l} \text{xx} \\ \text{(xx[,L])} \\ \text{(,L)} \end{array} \right\}$$

The two-character identifier, xx, is appended to IEAIPS to specify one of several possible parmlib members from which the system resources manager will obtain its parameters. If the L option is specified, the system displays the contents of the IEAIPSxx parmlib member at the operator's console as the system processes the member. (For additional information, see "IPS" in the description of the IEASYSxx member.)

Syntax Rules

See "Part 5: The System Resources Manager."

IBM-Supplied Default

IEAIPS00

Member Name: IEALPAxx (Modified LPA List)

Use of the Member

IEALPAxx contains the names of reenterable modules that NIP will load from SYS1.SVCLIB, the LNKLST concatenation, and the LPALST concatenation as a temporary extension to the existing pageable link pack area (PLPA). The extension is temporary in that the modules will remain in the paging data sets and be listed on the active LPA queue only for the duration of an IPL. They will *not* be automatically quick-started or warm-started (reinstated without re-specification of the MLPA parameter). Note that both the modified LPA (MLPA) and the fixed LPA (FLPA) modules (those named in IEAFIXxx) do not require a search of the LPA directory by the program manager when one of the modules is requested. The modified LPA, unlike the fixed LPA, however, contains *pageable* modules that behave in most respects like PLPA modules.

Note: Any modules specified for either MLPA or FLPA must come from authorized libraries. See the IEAAPF00 member. There may be non-authorized libraries present in the LNKLST concatenation.

You may use IEALPAxx to temporarily add or replace SVC or ERP routines. Another possible application would be the testing of replacement LPA modules that have been altered by PTFs.

LPA modules specified in IEALPAxx are placed in the MLPA or in the extended MLPA, depending on the RMODE of the modules. Modules with an RMODE of 24 are placed in the MLPA, while those with an RMODE of ANY are placed in the extended MLPA. By default, the MLPA and the extended MLPA are page protected, which means that a protection exception will occur if there is an attempt to store data into either area. To override page protection, use the NOPROT option on the MLPA system parameter.

LPA modules that have been replaced via IEALPAxx are not physically removed from the pageable LPA or from the LPA directory. They are, however, logically replaced because, when one of them is requested, the program manager searches the active LPA queue (a chain of CDEs), finds the name of the temporary replacement, and does not examine the LPA directory which contains the name of the replaced module.

The system searches the fixed LPA before the modified LPA for a particular module and selects the module from the modified LPA only if the module is not also in the fixed LPA.

If the first load module of a type 3 or 4 SVC routine is added or replaced, the SVC table is updated as required.

Note: An installation might want to change the device allocation structure for different job mixes or processor configurations. The MVS configuration program builds the eligible device table. See *MVSCP Guide* for information on the EDT.

To make NIP load the new tables for use by device allocation, use the LNKLSTxx parmlib member to concatenate SYS1.MLPALIB to SYS1.LINKLIB. When the operator replies to the SPECIFY SYSTEM PARAMETERS message with the appropriate SYSP=xx or MLPA=xx parmlib member suffix, the system uses the IEALPAxx member containing the new tables (via the LNKLST concatenation). Otherwise, NIP loads the device allocation tables from SYS1.LPALIB, and the system does not use the new tables.

The MVS configuration program builds the eligible device table. For more information on the EDT, see *MVSCP Guide*.

Parameter in IEASYSxx (or specified by the operator):

MLPA= { **aa** | **(aa[,L][,NOPROT])** | **(aa,bb,...[,L][,NOPROT])** }

The two alphameric characters, represented by aa (or bb, etc.), are appended to IEALPA to form the name of the IEALPAxx parmlib member(s). If the L option is specified, the system displays the contents of the IEALPAxx parmlib member(s) at the operator's console as the system processes the member(s).

Because the modified LPA is not a permanent addition to the LPA, you should probably specify MLPA in an alternate system parameter list (IEASYSxx) and not specify the parameter in IEASYS00. Alternately, you can have the operator enter the parameter from the console.

The LPA modules in the IEALPAxx parmlib member(s) are page protected in storage, by default. If an attempt is made to store into a page-protected module, a protection exception occurs. However, the NOPROT option allows an installation to override the page protection default. When NOPROT is specified, the LPA modules in the IEALPAxx parmlib member(s) are not page protected.

Syntax Rules

The following rules apply to the creation of IEALPAxx by means of the IEBUPDTE utility:

- On the first record, start with the data set name -- SYS1.SVCLIB, SYS1.LPALIB (meaning the LPALST concatenation), or SYS1.LINKLIB (meaning the LNKLST concatenation) -- from which modules will be taken. Follow the name with at least one blank.
- The allowable data set names are SVCLIB, LINKLIB and LPALIB.
- After the blank, list the modules to be loaded, separated by commas. Do not put a blank before or after a comma that separates module names on a record. You can use all columns except 72 through 80.
- Indicate continuation of the list by a comma after the last name on all records, except the last.
- Do not include the data set name on continued records.
- Place a *new* data set name at the beginning of a record, followed by at least one blank and a new string of module names.

- You can use major names or alias names, or both. Any alias names by which a module is to be accessed must be included. If an alias name is included and the associated major name is omitted, the system will locate the major name.
- End the last record with one or more blanks.

Syntax Example:

```
1st record     SYS1.LINKLIB    IKJPARS,IKJPARS2,IKJSCAN,
                               IKJEFD00,IKJDAIR
2nd record     SYS1.SVCLIB     IGC0009C,
3rd record     IGC09301,IGC09302,IGC09303
```

IBM-Supplied Defaults

None

Internal Parameters

Not applicable

Member Name: IEAOPTxx

Use of the Member

IEAOPTxx contains the parameters that affect swapping and other decisions made by the system resources manager (SRM). Installations with unique resource requirements can specify selected parameters in IEAOPTxx to replace the default values. Before making such a change it is, in most cases, advisable to study the related segments of code in order to predict the impact of the change on the SRM's evaluation and decision-making process. For more information, see "OPT Concepts" in "Part 5: The System Resources Manager."

Specifying the IEAOPTxx member is optional. Likewise, the specification of any parameter within the member is optional.

Parameter in IEASYSxx (or specified by the operator):

OPT = { **xx** | **(xx[,L])** | **(,L)** }

The two-character identifier, xx, is appended to IEAOPT to specify one of several possible parmlib members that can contain parameters used by various algorithms of the system resources manager. If the L option is specified, the system displays the contents of the IEAOPTxx parmlib member at the operator's console as the system processes the member. (For additional information, see "OPT" in the description of the IEASYSxx member.)

Syntax Rules

See "Part 5: The System Resources Manager."

IBM-Supplied Default

IEAOPT00

Member Name: IEAPAKxx (LPA Pack List)

Use of the Member

IEAPAKxx is an installation-supplied member that contains groups of names of modules in the LPALST concatenation that are executed together or in sequence. (As an example, the load modules of a type-4 SVC routine are called sequentially and executed as a group.) The member is used only during a "cold" start (CLPA specified), when the PLPA is loaded from the LPALST concatenation.

NIP uses the specified IEAPAKxx member(s) to determine the order in which modules are to be loaded from the LPALST concatenation into the pageable LPA. These modules are packed together, if possible, on a single page. The purpose is to reduce page faults. The LPA can greatly contribute to page faults because it is highly used. The IEAPAKxx list can significantly reduce page faults.

Each group ideally should not exceed 4K bytes in size. If a group exceeds 4K, the module in the group that causes 4K to be exceeded, and all later modules in the same group, will be loaded at the next page boundary in the LPA. In contrast, NIP loads other modules (those not listed in IEAPAKxx) in size order, the largest modules first, then the smaller modules. Unused spaces within page boundaries are filled, if possible, with modules smaller than 4K.

You should select link pack area programs for inclusion in the system pack list to reduce the number of page faults from the pageable link pack area and thereby enhance system performance. The affinity of programs for each other and the size of the programs determine which should be selected for a pack list entry. Program affinity means that one program will usually refer to another program when the first program is invoked. By putting programs that refer to each other into the same pack list entry, and thereby on the same page, extra page faults are avoided, because the programs are always in real storage together.

Very large programs that are to be put into the link pack area should be link edited so that CSECTs that have affinity[6] for other CSECTs are placed within the same page. The linkedit ORDER statement can be used to group CSECTs into pages. This process will accomplish the same goal as the pack list entries, because page faults during execution will be reduced.

Note: An installation can create one or more IEAPAKxx parmlib members, as needed. The system uses the IEAPAK00 member (if it exists) if PAK=xx is not included in the IEASYSxx parmlib member or is not specified by the operator. However, initialization continues whether or not there is an IEAPAK00 or IEAPAKxx member.

---

[6] The modules either execute at the same time or call each other.

Parameter in IEASYSxx (or specified by the operator):

$$\text{PAK} = \left\{ \begin{array}{l} \text{aa} \\ \text{(aa,bb,...[,L])} \\ \text{(,L)} \end{array} \right\}$$

The two alphameric characters, represented by aa (or bb, etc.), are appended to IEAPAK to form the name of the IEAPAKxx parmlib member(s). If the 'L' option is specified, the system displays the contents of the IEAPAKxx member(s) at the operator's console when the system processes the member(s).

Syntax Rules

The following rules apply to the creation of IEAPAKxx by means of the IEBUPDTE utility:

- The member consists of "groups" or entries containing load module names. Each group is enclosed in parentheses. For example: (IGG019CM,IGG019CN,IGG019CO,IGG019CP,IGG019CR).
- All modules in a group must have the same residence mode (RMODE).
- Separate modules within each group by commas.
- Separate each group from the next by a comma after the closing parenthesis.
- Do not use alias module names. NIP processes only major names.
- All named modules must be refreshable because LPA modules must have this attribute.

IBM-Supplied Default

None

Internal Parameters

Not applicable

Member Name: IEASLPxx

Use of the Member

IEASLPxx is an installation-supplied member of SYS1.PARMLIB that can contain any valid SLIP command. The commands can span multiple lines, and the system processes the commands in order.

You can move any SLIP commands in the COMMNDxx and IEACMDxx parmlib members into a IEASLPxx parmlib member. If you do, you must replace the SLIP commands in the appropriate parmlib members, COMMNDxx and IEACMDxx, (where xx is the COMMNDxx and IEACMDxx member ids) with the following line:

```
COM='SET SLIP=xx'
```

where xx is the IEASLPxx member id.

See the COMMNDxx and IEACMDxx parmlib members for related information.

Parameter in IEASYSxx (or issued by the operator):

None

Syntax Rules

The following rules apply to the creation of IEASLPxx.

- Begin each command on a new line.
- Data, including comments, must be contained in columns 1-71; the system ignores columns 72-80.
- Comments must begin with an asterisk "*" in column one.
- Enter the SLIP commands as you would enter them on the operator's console.
- Do not enclose the SLIP commands with double or single quotes or extra keywords.
- You may code multiple lines to specify a SLIP command.
- After the SET, MOD, or DEL keyword, the first blank ends the SLIP processing for that line; the system ignores all subsequent characters on that line.

The following is an example of a SLIP command within an IEASLPxx member:

```
SLIP SET,C=C06,
     A=WAIT,END
```

IBM-Supplied Defaults

None

Using System Commands

When you issue SET SLIP=xx, the system processes all of the commands in IEASLPxx.

To view the status of the SLIP traps, use the DISPLAY SLIP command. See *System Commands* for more information on the SET and DISPLAY commands.

Member Name: IEASVCxx

Use of the Member

IEASVCxx is an installation-supplied member that contains the information an installation supplies to define its own SVCs. These SVCs can be numbered from 200 through 255. NIP processing uses the member(s) to place the specified SVCs in the SVCTABLE, which resides in the extended read only nucleus.

Parameter in IEASYSxx (or issued by the operator):

SVC= { **(aa)** | **(aa,bb,...L)** }

The two alphameric characters, (aa, bb, etc.) are appended to IEASVC to form the name of the parmlib member(s).

Note: The contents of IEASVCxx appears on the operator's console if the (,L) option is specified with either the SVC= keyword or in response to the 'SPECIFY SYSTEM PARAMETERS' prompt.

Syntax Rules

The following rules apply to the creation of IEASVCxx.

- Data, including comments, must appear in columns 1 through 71. Do not use columns 72-80 for data; these columns are ignored.
- Comments must begin with "/*" and end with "*/"
- One or more blanks may precede the SVCPARM statement.
- One or more blanks may follow the SVCPARM statement.
- The system processes each statement in a member sequentially.
- If you use multiple SVCPARM statements for the same SVC number, the first valid statement is processed. The rest are ignored.
- SVCNUM, REPLACE, and TYPE are required parameters for every statement.
- SVCNUM, REPLACE, and TYPE are positional parameters. SVCNUM must be coded as the first parameter with REPLACE following as the second parameter.

Syntax Examples

The following 2 examples show hold to code for a type 1 and a type 3 SVC.

```
SVCPARM
245,REPLACE,TYPE(1),EPNAME(SVC245),LOCKS(CMS),APF(YES)

SVCPARM 244,REPLACE,TYPE(3),EPNAME(IGC0024D),LOCKS(LOCAL)
```

IBM-Supplied Defaults

None

Internal Parameters

Statement	Parameter	Meaning and Use	Default Value
SVCPARM	svcnum	Specifies a decimal number from 200 to 255. When a program issues an SVC instruction that contains this number, the system invokes the corresponding SVC routine.	
	,REPLACE	Specifies that an SVCTABLE entry is to be updated.	
	,TYPE(typenum)	Specifies the SVC type (1, 2, 3, 4, or 6) being defined.	
	,EPNAME(epname)	Specifies the entry point name of the SVC routine. The EPNAME parameter is optional unless an *epname* other than the IBM default SVC name is desired. Default names follow standard IBM SVC naming conventions. For example, type 1, 2, or 6 SVC routines will be named IGCnnn; nnn is the default number of the SVC routine. Type 3 and 4 SVC routines will be name IGC00nnn; nnn is the signed decimal number of the SVC routine. The *epname* must be a load module (or an alias) in LPA or link edited into the nucleus.	
	,LOCKS (lname,lname,...)	Specifies the name of the system lock(s) this SVC requires. lname can be LOCAL, CMS, SRM, SALLOC, or DISP. Type 1 SVCs do not need to specify the LOCAL lock as it is always obtained. Type 2, 3, and 4 SVCs specifying the CMS lock must also specify the LOCAL lock. Type 3 and type 4 SVCs may not specify any global spin lock. Type 6 SVCs may not specify any locks.	
	,APF { YES / NO }	Specifies whether or not the program invoking the SVC must be authorized. If you specify YES, the program issuing the SVC must be authorized. If you specify NO, the program issuing the SVC can be unauthorized.	NO
	,NPRMPT { YES / NO }	Specifies whether the SVC cannot be preempted (YES) or can be preempted (NO).	NO
	,AR { YES / NO }	Specifies whether or not the SVC can be issued in access register ASC mode.	NO

Member Name: IEASYSxx (System Parameter List)

Use of the Member

The system programmer specifies system parameters through a combination of three sources: the DATASET macro at sysgen, IEASYSxx members of parmlib, and operator responses to the SPECIFY SYSTEM PARAMETERS message. Some system parameters can be specified through sysgen macros and automatically copied to the IEASYS00 member of parmlib. The PAGEDSN keyword of the DATASET macro is mandatory at sysgen. Other parameters can be placed in the IEASYS00 member or in an alternate system parameter list (IEASYSxx), to provide a fast initialization with little or no operator intervention.

Figure 3-6 shows which sysgen parameters are always placed in IEASYS00, which parameters are optionally placed in IEASYS00, and which NIP parameters are equivalent to these sysgen parameters.

Figure 3-6. SYSGEN Parameters that Are Copied to the IEASYS00 Member

SYSGEN Macro	Equivalent Initialization	Related Parmlib Member	
DATASET			
DUPLEXDS,NAME=	DUPLEX=		Parameter is copied to IEASYS00 only if specified at sysgen.
PAGEDSN=	PAGE=		The specification of at least three DATASET macros with the PAGEDSN keyword is required at sysgen. This specification causes sysgen to place the PAGE parameter and the specified dsnames into IEASYS00. The dsnames specified with PAGEDSN can be changed or added to at IPL.
RESIDNT=	FIX=00	IEAFIX00	FIX=00 is copied to IEASYS00. The member name specified in the RESIDNT keyword is placed in IEAFIX00.
SWAPDSN=	SWAP=		Parameter is copied to IEASYS00 only if specified at sysgen.

The installation has the choice of placing all or some system parameters in the IEASYS00 member, and other parameters, or other values of the same parameters, in one or more alternate IEASYSxx members. IEASYS00 is the likely place to put installation defaults or parameters that will not change from IPL to IPL. The system programmer can add to or modify sysgen-created parameters in this member by means of the IEBUPDTE utility. The alternate IEASYSxx member(s), in contrast, should contain parameters that are subject to change, possibly from one work shift to another.

Use of the IEASYS00 member can minimize operator intervention at IPL. Since IEASYS00 is read automatically, the operator can respond to SPECIFY SYSTEM PARAMETERS by replying ENTER or 'U', and need not enter parameters unless an error occurs and prompting ensues.

If a parameter list other than IEASYS00 is desired at a particular IPL, the operator must indicate the alternate list by replying SYSP=xx in response to the SPECIFY SYSTEM PARAMETERS message. (See the SYSP parameter description in this section for additional information.)

The use of system parameter lists in parmlib thus offers two main advantages:

- They shorten and simplify the IPL process by allowing the installation to preselect system parameters.
- They provide flexibility in the choice of system parameters. The operator can specify SYSP=xx to select one of several alternate lists.

The system parameters that sysgen places in IEASYS00 (shown in Figure 3-6) are not adequate to initialize the system; additional parameters are needed.

Overview of IEASYSxx Parameters

The following list briefly defines all system parameters that can be placed in an IEASYSxx or IEASYS00 member (or specified by the operator). Detailed discussions of these parameters are provided in later sections of the IEASYSxx topic.

Note: PAGE is the only mandatory parameter that has no internal default. PAGE must be specified.

The GRSRNL parameter is mandatory when the GRS parameter is specified as JOIN or START.

Parameter	Use of the Parameter
APF	Names the parmlib member (IEAAPFxx) that contains authorized data set names.
APG	Specifies the priority value of the automatic priority group for use by the system resource manager, if one is not specified in the selected IPS.
CLOCK	Completes the name of the parmlib member (CLOCKxx) that prompts the operator to initialize the TOD clock during NIP and specifies the difference between the local time and Greenwich Mean Time (GMT).
CLPA	Tells NIP to load the link pack area with the modules contained in the LPALST concatenation. It also purges VIO data set pages that were used in the previously initialized system. Thus, CLPA implies CVIO.
CMB	Specifies the I/O device classes for which measurement data is to be collected, in addition to the DASD and tape device classes.
CMD	Completes the name of the parmlib member (COMMNDxx) that contains commands to be issued internally during master scheduler initialization.
CON	Completes the name of the parmlib member (CONSOLxx) that centralizes control of the console configuration for your installation. CONSOLxx also contains the initialization values for communication tasks, the characteristics for the hardcopy log, and default routing codes for messages that do not have routing information.
CSA	Specifies the sizes of the virtual common service area and extended common service area in multiples of 1K bytes.
CVIO	Deletes previously used VIO data set pages from the paging space. This parameter is automatically included when CLPA is specified.
DUMP	Specifies whether SYS1.DUMP data sets for SVC dump are to be on direct access device(s) or tape devices. This parameter can also indicate that *no* SYS1.DUMP data sets are to be made available for SVC dumps.
DUPLEX	Specifies a duplex page data set name or overrides the existing duplex data set name. This parameter is ignored on quick starts and warm starts (non-CLPA IPLs).
FIX	Completes the name of one or more parmlib members (IEAFIXxx) that contain names of modules from SYS1.SVCLIB, the LNKLST concatenation, and the LPALST concatenation that are to be placed in a fixed LPA that lasts for the duration of the IPL.
GRS	Specifies whether the system is to start a global resource serialization complex, join an existing complex, or not be part of a complex.
GRSCNF	Completes the name of the parmlib member (GRSCNFxx) that contains the information needed to initialize a system that is to be part of a global resource serialization complex.
GRSRNL	Completes the name of one or more parmlib members (GRSRNLxx) that contain resource name lists (RNLs).
ICS	Completes the name of the parmlib member (IEAICSxx) that contains the installation control specification. The installation control specification is used by the system resources manager (SRM) to assign performance groups.
IOS	Completes the name of the parmlib member (IECIOSxx) that contains missing interrupt handler (MIH) statements used to modify MIH time intervals and HOTIO statements used to modify recovery actions specified in the hot I/O detection table (HIDT).
IPS	Completes the name of the parmlib member (IEAIPSxx) from which the system resources manager (SRM) will obtain the installation performance specification.
LNK	Completes the name of one or more parmlib members (LNKLSTxx) that contain names of data sets that are to be concatenated to SYS1.LINKLIB.
LNKAUTH	Specifies whether all data sets in the LNKLST concatenation are to be treated as APF authorized or whether only those that are named in the APF table are to be treated as APF authorized.
LOGCLS	Specifies the JES output class for the log data sets.
LOGLMT	Specifies the maximum number of WTLs (messages) for a log data set. When the limit is reached, the data set is scheduled for sysout processing.
LPA	Completes the name of one or more parmlib members (LPALSTxx) that contain names of data sets that are to be concatenated to SYS1.LPALIB for the purpose of building the pageable LPA (PLPA and extended PLPA).
MAXUSER	Specifies a value that the system uses (along with the RSVSTRT and RSVNONR parameter values) to build the address space vector table (ASVT). The system uses the ASVT to locate the various address space control blocks.

Figure 3-7 (Part 1 of 2). Overview of IEASYSxx Parameters

Parameter	Use of the Parameter
MLPA	Completes the name of one or more parmlib members (IEALPAxx) that names modules from SYS1.SVCLIB, the LNKLST concatenation, and the LPALST concatenation that are to be placed in a modified LPA that lasts for the duration of the IPL.
MSTRJCL	Completes the name of MSTJCLxx, a module in SYS1.LINKLIB, that contains the JCL used to start the master scheduler address space.
NONVIO	Designates one or more local page data sets that are not to be used for VIO paging.
NSYSLX	Specifies the number of slots in the system function table to be reserved for system linkage indexes (LXs).
OPI	Indicates whether the operator is to be allowed to override particular parameters, or all parameters, contained in IEASYSxx.
OPT	Completes the name of a parmlib member (IEAOPTxx) that contains parameters to be used by various algorithms of the system resources manager.
PAGE	Gives the names of new page data sets to be used as additions to or replacements for existing page data sets. The first-named data set is used for the PLPA and extended PLPA pages. The second-named data set is used for MLPA and CSA. The third and all subsequently named data sets are used as local page data sets. Replacement is possible only if the parameter is placed in IEASYSxx, and the operator selects this member by entering SYSP=xx. The PAGE parameter, when specified by the operator, can only add temporarily (until the next cold or quick start) to a parmlib page data set list.
PAGTOTL	Specifies the total number of page and swap data sets that can be allocated for the life of the IPL.
PAK	Completes the name of one or more parmlib members (IEAPAKxx) that contain groups of names of modules in the LPALST concatenation that are executed together or in sequence.
PURGE	Demounts all mass storage system volumes.
RDE	Specifies that the reliability data extractor feature is included. For information on RDE, see *SYS1.LOGREC Error Recording*.
REAL	Specifies the maximum amount of real storage, in 1K blocks, that can be allocated for concurrent ADDRSPC=REAL jobs.
RER	Specifies that the reduced error recovery procedures for magnetic tapes are in effect if they are included on the OPTCD parameter of a data definition (DD) statement or on the DCB macro instruction. If the DD statement or the DCB macro does not specify the reduced error recovery procedures, all requests for them are ignored.
RSU	Specifies the number of storage units to be made available for storage reconfiguration.
RSVNONR	Specifies the number of ASVT entries to be reserved for replacing those entries marked non-reusable for the duration of an IPL.
RSVSTRT	Specifies the number of address space vector table (ASVT) entries to be reserved for address spaces created in response to a START command.
SCH	Specifies a parmlib member (SCHEDxx) from which the master scheduler will obtain its parameters. This member centralizes control over the eligible device table, the size of the master trace table, the program properties table, and the completion codes that are eligible for automatic restart.
SMF	Specifies a parmlib member (SMFPRMxx) from which SMF will obtain its parameters.
SQA	Gives the number of 64K blocks of virtual system queue area to be created at IPL (in addition to the system's minimum virtual SQA and extended SQA).
SSN	Completes the name of the parmlib member, IEFSSNxx, that contains the information to be used in identifying subsystems that are to be initialized.
SVC	Completes the name of the parmlib member (IEASVCxx) that contains the information an installation supplies to define its own SVCs. NIP processing places these SVC in the SVCTABLE.
SWAP	Specifies the names of new swap data sets to be used as additions to or replacements for existing swap data sets.
SYSNAME	Specifies the name of the system being initialized. Global resource serialization uses this name to identify the different systems in a complex.
SYSP	Specifies one or more alternate system parameter lists (IEASYSxx) that are to be read by NIP in addition to IEASYS00. SYSP may be specified only by the operator.
VAL	Names one or more parmlib members (VATLSTxx) that contain "mount" and "use" attributes of direct access devices.
VRREGN	Gives the default real-storage region size for an ADDRSPC=REAL job step that does not have a REGION parameter in its JCL.

Figure 3-7 (Part 2 of 2). Overview of IEASYSxx Parameters

Changes to Initialization Parameters

For a list of parameters that have changed or that are no longer supported by the current level of MVS, refer to *MVS/Extended Architecture Conversion Notebook.*

Parameter Specified by the Operator:

SYSP=xx

The operator specifies this parameter to indicate an alternate system parameter list to be used by NIP in addition to IEASYS00. (See SYSP in "Internal Parameters" later in this section.)

Syntax Rules

The following rules apply to the creation of IEASYSxx:

- Use columns 1 through 71 of each record for parameters. Leading blanks are acceptable.
- Do not use columns 72 through 80, since NIP ignores these columns.
- Separate parameters with commas.
- Indicate continuation by a comma, followed by at least one blank.
- Begin subsequent records in any column.
- Enclose multiple subparameters in parentheses. The number of subparameters is unlimited.

Syntax Example:

```
...DUMP=(TA,282),MLPA=(00,01,02,03,L)
```

IBM-Supplied Defaults

The default member IEASYS00, initially created at sysgen, always contains PAGE. Particular parameters of the DATASET sysgen macro are copied to IEASYS00, if the installation specifies them during system generation.

Internal Parameters

The IEASYSxx parameters are listed alphabetically and are individually described. These parameters may optionally be issued by the operator, although such manual issuance would slow the IPL.

Note: "Value range," if applicable, means the syntactically acceptable range of values, not necessarily a range of values reasonable for function or performance. The "associated parmlib member" refers to the parmlib member that is named by the parameter. For example, IEAAPF01 is named by the APF=01 parameter in IEASYSxx, or entered by the operator.

APF=xx

Meaning and Use: The two alphameric characters xx are appended to IEAAPF to form the name of parmlib member IEAAPFxx. This member lists the data set names and volume serial numbers of authorized data sets. SYS1.LINKLIB and SYS1.SVCLIB are automatically included as authorized data sets. SYS1.LPALIB is automatically authorized only for the period during system initialization.

The installation creates the default parmlib member IEAAPF00.

Value Range: Not applicable

Default Value: SYS1.LINKLIB and SYS1.SVCLIB alone are authorized if APF=xx is not specified. However, if the default for the LNKAUTH system parameter is taken (LNKAUTH=LNKLST) or is specified in IEASYSxx or by the operator, libraries in the LNKLST concatenation are also authorized when accessed as part of the LNKLST concatenation. If a library is in the LNKLST concatenation, but is not APF authorized, referencing this library via a JOBLIB or STEPLIB DD statement will cause the library to be considered unauthorized for the duration of the job or step, respectively.

Associated Parmlib Member: IEAAPFxx

APG=nn

Meaning and Use: The one or two-digit number nn specifies the priority value of the automatic priority group (APG). Job steps that do not include the DPRTY keyword in their EXEC statements are automatically placed in the APG. DPRTY is specified as DPRTY=(value1,value2). Job steps for which value1=APG priority will also be in the APG group. Job steps for which value1 is greater than APG will have a higher priority than APG-group job steps. Job steps for which value1 is less than APG will have a lower priority than APG-group job steps. (Value2 is not used for APG. See *JCL Reference* for a description of value2.) Use the APG parameter of IEASYSxx in conjunction with the APG parameter of IEAIPSxx to determine dispatching priorities. The APG parameter of IEASYSxx sets the high-order digit of SRM-controlled dispatching priorities. The APG parameter of IEAIPSxx sets the low-order digit. The APGRNG and APG parameters of IEAIPSxx are mutually exclusive, and APGRNG overrides the APG parameter in IEASYSxx. For further information, see "Part 5: The System Resources Manager."

Value Range: 0-13

Default Value: 7

Associated Parmlib Member: None

CLOCK= { **aa** | **(aa,bb...L)** }

Meaning and Use: The two alphameric characters are appended to CLOCK to form the name of the CLOCKxx member of SYS1.PARMLIB. If you specify the L option in the syntax of the CLOCKxx member, the system writes all statements read from the CLOCKxx member to the operator's console. The member specifies whether to prompt the operator for during the TOD initialization or not and provides the difference between the local time and GMT.

Value Range: Any two alphameric characters.

Default Value: CLOCK=00

Associated Parmlib Member: CLOCKxx

CLPA (Create Link Pack Area)

(See also the MLPA parameter for temporary additions to the LPA, and the CVIO parameter for the deletion of VIO data sets.)

Meaning and Use: This parameter causes NIP to load the LPA with all modules contained in the LPALST concatenation. Modules listed in the specified LPA pack list member (IEAPAKxx) are packed together, preferably in one-page groups. (See description of IEAPAKxx.) Modules not in the pack list are loaded in size order, large modules first, then smaller modules to fill unused space.

PLPA pages are written to auxiliary storage. Only one set of PLPA pages can exist in paging space, although the same PLPA pages are duplexed on two separate page data sets for error recovery if a duplex data set is specified. Modules in the LPALST concatenation must be reenterable and refreshable because the system uses the processor's page protection facility, which enforces read-only access to each PLPA page.

CLPA should be specified after the installation has modified a data set in the LPALST concatenation and wants to reload the PLPA with new or changed modules.

Note: CLPA also implies CVIO, so that VIO data set pages on local page data sets are automatically purged. (See the description of CVIO for further information.)

The CLPA parameter is not needed at the *first* IPL. NIP detects the cold start condition internally, noting that the PLPA has not been loaded.

If CLPA is not specified, NIP tries to find a usable PLPA in the existing page data sets. If NIP is successful, a quick start or a warm start occurs, and the auxiliary storage manager (ASM) obtains the records that specify where the PLPA pages reside on auxiliary storage. It then reestablishes the previous set of PLPA pages. The old PLPA may be reused for any number of system initializations, as long as CLPA is not specified. However, page data sets that contain the last used set of PLPA pages must be mounted. If they are not, the operator is asked to mount them. If the operator bypasses mounting, ASM initialization requests a different page data set and forces a "cold" start. NIP then reestablishes the PLPA as it does when CLPA is specified. In this cold start, both the previously established PLPA *and* existing VIO data set pages are logically deleted from paging space.

The fixed LPA and the modified LPA, however, are not automatically reused in a quick start or a warm start. They must be respecified. Existing VIO data set pages on local page data sets are retained in a warm start, unless the CVIO or CLPA parameter is forced. Such pages are not retained in a quick start or a cold start. (See the description of the CVIO parameter.)

If CLPA is specified and a set of PLPA pages already exists on a paging data set, NIP frees the existing PLPA and updates the appropriate records to reflect the new PLPA pages on auxiliary storage. NIP loads the LPA from the LPALST concatenation, as previously described.

Value Range: Not applicable

Default Value: Not applicable

Associated Parmlib Member: None

CMB= { **option** | **(option,option[,option] . . .)** }

Meaning and Use: This parameter allows the installation to specify I/O device classes for which measurement data is to be collected, in addition to the DASD and tape device classes. (Measurement data is always collected for DASD and tape devices.) The system resources manager (SRM) uses the CMB parameter options to calculate the size of the channel measurement block (CMB), based on the number of devices that are generated for each class. For each I/O device to be monitored, SRM needs a 32-byte entry in the CMB, which it builds in contiguous real storage in the extended CSA. The size and location of the CMB

is fixed at initialization. That is, the CMB cannot be expanded or moved until the next IPL.

The channel stores measurement data in the CMB on a device basis. SRM uses the measurement data to perform its device selection and I/O load balancing functions.

The operator is notified if the CMB parameter contains a syntax error (message IEA926I) or if there is insufficient storage to accommodate measurement of the specified I/O device classes (message IEA340I). In both cases, SRM will prompt for respecification of the CMB parameter. If, after respecification, storage is still unavailable for measurement of the optional device classes, measurement will be done only for DASD and tape devices. Should storage also be unavailable for measurement of DASD and tape devices, SRM marks the measurement facilities as inoperative and informs the operator that device selection and I/O load balancing will be performed without I/O measurement data (messages IEA966I and IEA340I).

Operand Descriptions: One or more of the following options can be specified:

UNITR
specifies unit record devices

COMM
specifies communications equipment, including the channel-to-channel adapter (CTC)

GRAPH
specifies graphics devices

CHRDR
specifies character reader devices

n
specifies a number of CMB entries to be built in addition to those needed for DASD and tape devices and any other CMB parameter options specified. The number of additional entries is a decimal integer in the range 0 through 4096.

Examples of Valid CMB Statements:

```
CMB=UNITR
CMB=(COMM,100)
CMB=(UNITR,CHRDR,COMM,150)
CMB=50
```

Value Range: For the n operand, specify a decimal integer from 0 through 4096.

Default Value: None

Associated Parmlib Member: None

CMD= { **aa** | **(aa,bb...)** }

Meaning and Use: The two alphameric characters, represented by aa (or bb, etc.), specify one or more COMMNDxx members of parmlib. The installation can specify multiple members. Each member can contain automatic operator commands that the installation wants executed during master scheduler initialization. Examples of such commands are those that start GTF and TCAM. Job entry subsystem commands are not accepted because automatic commands are processed before the job entry subsystem (JES2 or JES3) is started.

If the CMD parameter is not specified, the COMMND00 member is used if it exists. If COMMND00 does not exist or cannot be read, initialization continues without any internally issued commands.

Value Range: Any two alphameric characters.

Default Value: CMD=00

Associated Parmlib Member: COMMNDxx (See description of this member.)

CON= { **aa** / **(aa,L)** }

Meaning and Use: This parameter is required at IPL. The two alphameric characters (aa) are appended to CONSOL to form the name of the installation-created CONSOLxx member of SYS1.PARMLIB. If you specify the L option, the system writes all of the statements read from the CONSOLxx member to the operator's console.

In CONSOLxx, you define the console configuration for your installation. In the parmlib member, you define the MCS consoles, the master console and up to 98 secondary consoles.

If the CON parameter is not specified, IPL can not continue.

Value Range: Any two alphameric characters.

Default Value: None

Associated Parmlib Member: CONSOLxx (See description of this member.)

CSA=(a,b)

Meaning and Use: This parameter specifies the sizes of the virtual common service area (CSA) and extended CSA in multiples of 1K (1024 bytes). The subparameter "a" specifies the size of the CSA, located below 16 megabytes. The subparameter "b" specifies the size of the extended CSA, located above 16 megabytes.

The specified size of the CSA is subtracted from the bottom of PLPA, after the bottom PLPA address is rounded down to the next 4K (page) boundary. If the resulting virtual address for the bottom of the CSA is not on a megabyte boundary, further rounding down occurs so that the bottom CSA address is on the next megabyte boundary.

Similarly, the specified size of the extended CSA is added to the top of the extended PLPA, after the top extended PLPA address is rounded up to the next 4K boundary. If the resulting virtual address for the top of the extended CSA is not on a megabyte boundary, further rounding up occurs so that the top extended CSA address is on the next megabyte boundary.

The CSA (including the extended CSA) is an address range in each address space that is used for common system functions (functions not related to a particular address space). For example, the system allocates buffers for LOG and SMF from the CSA. The CSA is duplexed for recoverability, as are the other common areas, PLPA and MLPA, when a duplex page data set is being used.

In selecting values for the CSA parameter, understand that the system's process of rounding to a one-megabyte boundary can cause up to one megabyte of storage from the private area to be allocated to the CSA. However, consider the following:

- If the virtual storage manager runs out of SQA, it will try to obtain space from the CSA.

- A large CSA size will reserve space for future LPA growth. Such growth would be hampered if users were allowed to obtain very large private areas. A large CSA specification effectively limits the maximum private area that a user job can acquire.

Note: If you allocate excessive amounts of CSA or SQA, the system generates a warning message, and you must respecify the CSA parameter. The system also generates a warning message when the size of the entire common area below 16 megabytes exceeds 8 megabytes.

Value Range: On the "a" subparameter, 0 -- 9999 (one to four digits) for the CSA. On the "b" subparameter, 0 --2080767 (one to seven digits) for the extended CSA.

Default Range: For each subparameter, 100 -- 1023 (depending on the amount of storage added because of rounding to a segment boundary).

Associated Parmlib Member: Not applicable

CVIO (Clear VIO)

Meaning and Use: This parameter specifies that all VIO data set pages on auxiliary storage are to be deleted from page space. A typical application would be the purging of VIO data set pages when the system is re-IPLed after a previous end-of-day (EOD).

Note: If you want the auxiliary storage manager (ASM) to purge and reinitialize *both* the VIO data set pages *and* the PLPA pages, specify CLPA. CLPA always implies CVIO.

If you do not specify CVIO, a warm start IPL occurs and VIO data set pages are retained for restart processing. (Such restart would be possible for some data sets after a temporary system failure.) ASM reestablishes the VIO data set pages that were checkpointed before the system failure. This action, of course, does not ensure that the job entry subsystem will reuse these data sets.

If one or more volumes that contain VIO pages are not mounted on a warm start IPL, ASM requests the operator to mount the missing volume(s). If the operator doesn't mount all the requested volumes that contain VIO pages, ASM deletes all VIO data set pages, just as if CVIO or CLPA had been specified. The operator receives a message that indicates that CVIO has been forced.

Value Range: Not applicable

Default Value: None

Associated Parmlib Member: None

DUMP= { **NO** | **DASD** | **(DASD,xx-yy)** | **L** | **(TA,devno1,devno2...)** }

Meaning and Use: This parameter specifies whether SYS1.DUMP data sets for SVC dump are to be on direct access device(s) or tape, and names the device number(s) if tape is to be used. Optionally, the parameter can specify (via NO) that no SVC dump data sets are to be made available. (The four options - DASD, (DASD,xx-yy), L, and TA - may be specified with a single DUMP parameter, if so desired.) SVC dump options are not included in parmlib. However, the installation can specify the options, if it so desires, by means of the CHNGDUMP operator command. When planning for dump data sets the

installation should be aware the dump data sets can sometimes contain privileged data. By using protected data sets (via passwords or other security methods), the installation can limit access.

Operand Descriptions:

NO
specifies that no dump data sets will be available for SVC dump.

Note: Dump data sets can be specified after IPL by using the DUMPDS command.

DASD
specifies that the currently cataloged SYS1.DUMPnn data sets (if any), on permanently resident direct access volumes, are to be used. The catalog will be scanned for SYS1.DUMP00 through SYS1.DUMP99. DASD is the default if the DUMP parameter is omitted.

(DASD,xx-yy)
specifies that the currently cataloged SYS1.DUMPnn data sets (if any), on permanently resident direct access volumes, are to be used. The catalog will be scanned for SYS1.DUMPxx through SYS1.DUMPyy, where xx and yy are decimal digits in the range 00 through 99.

Indicating which dump data sets are to be used by a particular system avoids unnecessary scanning of the possible 100 cataloged dump data sets and the possibility of more than one system using the same data sets.

L
is used with DASD or (DASD,xx-yy) to specify that cataloged SYS1.DUMPnn data sets are to be listed on the operator's console, with the status of empty or full. *Empty* means the data set is unused or is reusable; *full* means the data set is used and is ready to be printed. A message prompts the operator for the names of full data sets that he doesn't want to print. SVC dump will reuse these data sets. The operator can also specify in his reply the device numbers of additional tape units on which dump data can be written.

(TA,devno1,devno2...)
specifies that the tape unit with the specified device number is to be used as an SVC dump data set. If both this operand and the DASD operand are specified, the tape unit(s) are added to the available direct access dump data sets. If DASD is not specified, only the named tape units are made available.

Examples of Valid DUMP Statements:

```
DUMP=NO
DUMP=DASD
DUMP=(DASD,L)
DUMP=(DASD,L,(TA,282,283))
DUMP=(TA,282)
DUMP=(DASD,00-05)
DUMP=((DASD,10-20),L)
DUMP=((TA,580,581),(DASD,91-99),L)
```

How Dump Data Sets Are Used: Dump data sets may be all on tape, all on direct access devices, or on a combination of devices. Space for direct access data sets must be pre-allocated, and the data sets must be cataloged. Eligible device types consist of tapes supported by the system and any direct access device supported by the system that has a track size of at least 4104 bytes (4104 bytes equals 1 SVC

dump output record). The IBM 3850 Mass Storage System is not supported as a dump data set.

As many as 100 dump data sets may be allocated. Direct access data set names must be in the form SYS1.DUMPnn, in which nn may be digits 00 to 99.

Each direct access data set will contain only one dump. The installation should allocate several data sets that are large enough to contain the maximum size SVC dump expected. The size of a dump depends on the:

- SDATA and STOR options specified in the console DUMP command
- SDATA options specified in the CHNGDUMP command
- SDATA, STORAGE, LIST, LISTA, SUBPLST, KEYLIST, SUMLSTA, and SUMLIST options requested by the recovery routine issuing the SDUMP macro instruction
- Number of address spaces (ASIDs) to be dumped

For information on the commands, see *Operations: System Commands*; for information on the SDUMP macro, see *SPL: System Macros and Facilities.*

SYS1.DUMP data set size can be ***estimated*** with the following steps.

1. Calculate a private area
   Add the virtual address space system storage requirements to the approximate region size of the largest job or job step to be run.
2. Calculate a total private area estimate
   Multiple the private area estimate by the number of address spaces to be dumped with the SVC dump or DUMP command.
3. Calculate the number of bytes required by the SYS1.DUMP data set.
   Add the total private area to the system global virtual storage requirement (20% of the PLPA estimate)
4. Calculate the number of cylinders and tracks required on the device.
   Use the appropriate device characteristics and the record size for the dump records. SVC dump writes the dump in unblocked, 4104 byte records.

For example, on a 3330 direct access device, a large size SYS1.DUMP data set would require 60 cylinders There are 19 tracks per cylinder and 3 records per track. Therefore, there are a total of 1140 tracks (60 * 19 = 1140 tracks). With 3 records per track (3 records * 1140 tracks) this data set has space for 3420 SVC dump records. These records would contain approximately 13680K of storage. (Each record has an 8 byte header.)

Requirements for Dump Data Sets: Dump data sets must meet the following requirements:

- A data set can reside on only one volume; that is, a data set can't span across two or more volumes.
- Each direct access data set can contain only one dump.
- Direct access data sets must be on permanently resident volumes and be allocated, and cataloged. (For further information on the permanently resident attribute, see the description of the VATLSTxx member of parmlib.)
- The SPACE parameter on DD statements for direct access data sets must specify CONTIG and must not specify a secondary quantity.

- Dump data sets cannot be shared.
- Dump data sets on DASD must be initialized with an EOF record as the first record.

Note: The DUMPDS operator command makes it possible to add, delete, or clear dump data sets without having to re-IPL the system. (See the description of the DUMPDS command in *System Commands* and in *Dump and Trace Services.*

Processing of Dump Data Sets: The status and type of each dump data set is maintained in an internal queue by SVC dump. The data sets are processed in the order in which they are specified as operands of the DUMP parameter. If, for example, the parameter statement specifies DUMP=(TA,283),DASD,(TA,285), the first queue element would be for (TA,283). This would be followed by queue elements for cataloged DASD data sets on permanently resident volumes, in the order SYS1.DUMP00 to SYS1.DUMP99. Only 100 data sets are processed. The last queue element is for (TA,285), unless the limit of 100 data sets has already been reached.

Each queue element reflects the data set status, empty or full. An empty data set is available for use by SVC dump. A full data set can either be printed or optionally reused for a new dump. The data set is reused if the operator replies to message IEA877A by entering the data set name or tape device number. For example, the operator replies to message IEA877A:

R 00,DA=(xx,yy),(TA,ttt,uuu).

The characters xx,yy are the last two digits of two direct access data sets, named SYS1.DUMPxx and SYS1.DUMPyy. They are currently full but are to be reused. The characters ttt,uuu are the device numbers of two additional tape drives that are to be used for dump data sets.

The criteria that determines whether a dump data set is empty or full depends on the device type, DASD or tape. A tape is always considered empty. A DASD data set is empty only if the first record is "end of data." Otherwise, the data set is considered full.

SVC dump examines the data sets, as listed in the queue, in sequence. If a data set is empty, SVC dump uses it and marks the queue element accordingly. If a data set appears full, it is marked as being used and is skipped. (SVC dump always selects an empty DASD data set for its use before selecting a tape.) When all data sets have been marked in use, SVC dump reads the first record of each DASD data set to see if it has been emptied (printed). If so, the first record is "end of data." SVC dump updates the queue to reflect the current status, then reexamines the data sets, starting at the beginning of the queue.

Value Range: Not applicable

Default Value: DASD

Associated Parmlib Member: None

DUPLEX=dsname

Meaning and Use: The DUPLEX parameter specifies a paging data set to be used to hold a secondary copy of all common area system pages (all pages written to the PLPA or common page data sets). Only one duplex data set can be specified. The DUPLEX parameter is valid only on a cold start IPL (CLPA).

On either quick start or warm start IPLs, ASM ignores the DUPLEX parameter and attempts to use the same duplex page data set specified on the cold start IPL. If no duplex page data set was then specified, or if it is otherwise unavailable, duplexing is suspended. This parameter follows all of the normal override rules for system parameters.

How the Duplex Page Data Set Is Specified: On a CLPA IPL, the duplex page data set is specified from one of two sources: 1) IEASYS00 or IEASYSxx, or 2) the operator-specified DUPLEX parameter. On a non-CLPA IPL, the duplex page data set is specified from only a single source: the temporary page activity reference table (TPARTBLE). An outline of these specifications is as follows:

- **CLPA IPLs**
  - *The DUPLEX parameter in IEASYS00:* This data set name is specified along with the DUPLEXDS[7] keyword of the DATASET macro during system generation.
  - *The DUPLEX parameter in IEASYSxx*, an alternate system parameter list: If the operator selects this list (by using the SYSP parameter), the DUPLEX parameter in IEASYSxx overrides the DUPLEX parameter in IEASYS00.
  - *The DUPLEX parameter specified by the operator in the current IPL:* This DUPLEX parameter specification is *not* merged with that specified by IEASYSxx, as is the case with the PAGE parameter specification, but overrides those specifications. The operator specification lasts until the next cold start IPL.
- **Non-CLPA IPLs**
  - *TPARTBLE:* The temporary page activity reference table occupies the first 40K of the PLPA page data set. It contains the names of previously specified page data sets - including the name of the previous duplex page data set - that will be used for the current IPL.

Before an IPL, the page data set specified through the DUPLEX parameter, or through the sysgen DATASET[8] macro, must have been allocated, cataloged in the system master catalog, and preformatted by VSAM. This data set is a normal page data set used for duplexing. The data set can be formatted by using the DEFINE PAGESPACE processor of access method services as described in *Access Method Services Reference*.

Syntax Example for the DUPLEX Parameter:

```
DUPLEX=dsname
```

When specified on a cold start IPL, this statement causes ASM to use the specified data set for secondary (backup) copies of the PLPA, extended PLPA, and common area pages.

Note that neither UNIT nor VOLSER is specified. Because all data sets used as paging data sets must be cataloged, ASM initialization does not need externally

---

[7] The DUPLEXDS keyword causes the DUPLEX parameter to be placed in IEASYS00.

[8] For information about creating page data sets with the sysgen DATASET macro, refer to *System Generation Reference*.

specified volume serial numbers. The operator can either pre-mount the necessary volumes or wait for a mount message.

When the duplex page data set is full, ASM issues a message stating that duplexing has been suspended. Once suspended, duplexing can only recommence after a subsequent IPL. (See "Part 4: Auxiliary Storage Management Initialization" for space requirement guidelines

Value Range: Not applicable

Default Value: None

Associated Parmlib Member: None

FIX= { **aa** | **(aa[,L][,NOPROT])** | **(aa,bb...[,L][,NOPROT])** }

Meaning and Use: This parameter specifies one or more IEAFIXxx members of parmlib. The two alphameric characters, represented by aa (or bb, etc.), are appended to IEAFIX to name the member(s). If the L option is specified, the system displays the contents of the IEAFIXxx parmlib member(s) at the operator's console as the system processes the member(s).

The member(s) contain names of modules from SYS1.SVCLIB, the LPALST concatenation, and the LNKLST concatenation that are to be fixed in real storage as a fixed LPA. The fixed LPA modules are active only for the duration of an IPL, and will not be automatically reinstated by a quick start or a warm start IPL. You must respecify the FIX parameter in subsequent IPLs if you want to reinstate the fixed LPA.

The LPA modules that are fixed in storage are also page protected, by default. If an attempt is made to store into a page-protected module, a protection exception occurs. However, an installation can use the NOPROT option to override the page protection default. When NOPROT is specified, the LPA modules in the IEAFIXxx parmlib member(s) are not page protected in storage. If you specify the NOPROT option, you must also specify the L option.

Note: The FIX parameter may also be specified by using the RESIDNT keyword of the DATASET macro at sysgen. In this case, FIX=01 is automatically placed in member IEASYS00, and the modules specified by the RESIDNT keyword are listed in parmlib member IEAFIX01. (For additional information on the FIX option, refer to the description of the IEAFIXxx member. For information on the modified LPA option, see the descriptions of the MLPA parameter and the IEALPAxx member.)

Value Range: Any two alphameric characters.

Default Value: FIX=01, if the installation specified the RESIDNT keyword of the DATASET macro at sysgen.

Associated Parmlib Member: IEAFIXxx

GRS= { **JOIN** | **START** | **NONE** }

Meaning and Use: This parameter specifies whether or not the system being initialized is to participate in a global resource serialization complex. GRS=NONE indicates that the system is not to coordinate global resource requests with other systems in a global resource serialization complex. GRS=START and GRS=JOIN, however, indicate that the system is to coordinate global resource requests with other systems in a global resource serialization complex.

Specifying GRS=START indicates that the system is to start a global resource serialization complex and honor the requests of other systems (those systems that specify GRS=JOIN) to join the complex. Specifying GRS=JOIN indicates that the system is to join an existing complex of active global resource serialization systems and also honor the requests of other systems to join the complex. The system specifying GRS=START and a system specifying GRS=JOIN must be connected by at least one CTC adapter in order for a complex of two systems to be initialized.

Note: If you have specified either GRS=JOIN or GRS=START, then you must specify the GRSRNL parameter. You may also need to place certain resource names in the SYSTEMS EXCLUSION resource name list (RNL) in order to IPL the system. For example, if an initialization routine issues a global reserve on a resource, place the name of that resource in the SYSTEMS EXCLUSION RNL. For more information on the resources that you should place in the SYSTEMS EXCLUSION RNL, refer to the "RNL Considerations" topic in *Global Resource Serialization.*

Value Range: Not applicable

Default Value: GRS=JOIN

Associated Parmlib Member: None

GRSCNF=xx

Meaning and Use: This parameter identifies the GRSCNFxx parmlib member to be used to initialize a system in the global resource serialization complex. The two alphanumeric characters, xx, are appended to GRSCNF to form the name of the parmlib member, GRSCNFxx. Only one suffix can be supplied. The installation creates the member and places it in parmlib via IEBUPDTE. GRSCNF=xx is ignored if GRS=NONE is specified for the system during its initialization.

Value Range: Any two alphameric characters.

Default Value: GRSCNF00 is the parmlib member selected if you do not specify GRSCNF=xx. GRSCNF=00 is the default value in the IEASYS00 parmlib member.

Associated Parmlib Member: GRSCNFxx

GRSRNL= { **aa** | **(aa,bb...)** }

Meaning and Use: This parameter specifies one or more GRSRNLxx parmlib members. The two alphameric characters, represented by aa (or bb, etc.), are appended to GRSRNL to form the name of the GRSRNLxx member(s).

The GRSRNLxx member(s) contain resource name lists (RNLs). The system (specifically, global resource serialization) uses the RNLs to determine how to treat a resource that the installation defined in an RNL.

Notes:

1. If GRS=START or GRS=JOIN is specified during system initialization, the GRSRNL parameter is required.
2. If GRS=NONE is specified during system initialization, the system ignores the GRSRNL parameter.

Value Range: Any two alphameric characters.

Default Value: None

Note: GRSRNL=00 is placed in the IBM supplied IEASYS00 member built at sysgen. Otherwise, the GRSRNL parameter must be explicitly supplied.

Associated Parmlib Member: GRSRNLxx

ICS= { **xx** | **(xx[,L])** | **(,L)** }

Meaning and Use: This parameter specifies the parmlib member that contains the installation control specification. The installation control specification associates units of work (transactions) with performance groups. The two alphameric characters (xx) are appended to IEAICS to form the name of an IEAICSxx member of parmlib. If the L option is specified, the system displays the contents of the IEAICSxx parmlib member at the operator's console as the system processes the member.

If a member contains invalid specifications or cannot be found, SRM prompts the operator to specify an alternate member by respecifying ICS=xx. If the operator cancels the parameter by replying with the ENTER key, no installation control specification is used, and performance groups are assigned by the JCL or LOGON PERFORM parameter.

The operator can select a new installation control specification (that is, indicate that the system is to run under the control of an alternate IEAICSxx member) between IPLs by issuing the SET ICS command. For more information see "The SET ICS Command" in "Part 5: The System Resources Manager."

Value Range: Any two alphameric characters.

Default Value: None. If an installation control specification is not provided, performance groups are assigned by the JCL or LOGON PERFORM parameter.

Associated Parmlib Member: IEAICSxx

IOS=

Meaning and Use: This parameter specifies the parmlib member that contains (1) time intervals to be used by the missing interrupt handler (MIH) when scanning UCBs for missing interrupt conditions and (2) device threshold values and recovery actions to be used in the detection of, and recovery from, hot I/O conditions. The two alphameric characters (xx), specified in the parameter, are appended to IECIOS to form the parmlib member IECIOSxx.

Value Range: Any two alphameric characters

Default Value: None

Note: The I/O supervisor assumes default options. For a list of these defaults and their explanations see the description of the IECIOSxx parmlib member.

Associated Parmlib Member: IECIOSxx

IPS= { **xx** | **(xx[,L])** | **(,L)** }

Meaning and Use: This parameter specifies the particular installation performance specification (IPS) that will be used by the system resources manager.

The two alphameric characters, represented by xx, are appended to IEAIPS to form the name of an IEAIPSxx member of parmlib. If the L option is specified, the system displays the contents of the IEAIPSxx parmlib member at the operator's console as the system processes the member

If the member can't be found or contains invalid specifications, the initialization routine of the system resources manager prompts the operator to specify an alternate member, i.e., respecify IPS=xx. If the operator chooses to reply with the ENTER key, the system resource manager tries to use the default member IEAIPS00. If IEAIPS00 is invalid or unavailable, the system resource manager uses an internal set of IPS values, called the "skeleton IPS." The skeleton IPS avoids service rate distinctions among any jobs. It is merely a stopgap IPS intended to permit the completion of IPL.

The operator can select a new IPS (that is, indicate that the system is to run under the control of an alternate IEAIPSxx member) between IPLs by issuing the SET IPS command. For more information see "The SET IPS Command" in "Part 5: The System Resources Manager."

Value Range: Any two-character alphameric combination.

Default Value: IPS=00. The IEAIPS00 default member, supplied by IBM, can be modified by the installation. The use and contents of this default member are described under "Default IPS" in "Part 5: The System Resources Manager."

LNK= { **aa** | **(aa,bb,...[,L])** | **(,L)** }

Meaning and Use: This parameter specifies the LNKLSTxx member(s) of parmlib. The two alphameric characters, represented by aa (or bb, etc.), are appended to LNKLST to form the name of the LNKLSTxx member(s). If the L option is specified, the names of the data sets that are concatenated to SYS1.LINKLIB are displayed at the operator's console as the data sets are opened.

The LNKLSTxx member(s) list data sets that are to be concatenated to SYS1.LINKLIB. (For information on the use, contents, and syntax of this member, see the description of member LNKLSTxx.)

Value Range: Any two alphameric characters.

Default Value: LNK=00, causing selection of LNKLST00.

Associated Parmlib Member: LNKLSTxx

If the default for the LNKAUTH system parameter is taken, (LNKAUTH=LNKLST) or is specified in IEASYSxx or by the operator, libraries in the LNKLST concatenation are also authorized when accessed as

part of the LNKLST concatenation. If a library is in the LNKLST concatenation, but is not APF-authorized, referencing this library via a JOBLIB or STEPLIB DD statement causes the library to be considered unauthorized for the duration of the job or step.

LNKAUTH= {**LNKLST** | **APFTAB**}

Meaning and Use: This parameter specifies whether all data sets in the LNKLST concatenation are to be treated as APF-authorized when accessed as part of the concatenation or whether only those data sets named in the APF table are to be treated as APF-authorized.

Value Range: Not applicable

Default Value: LNKLST, meaning that all of the data sets in the LNKLST concatenation are to be treated as APF-authorized when accessed as part of the concatenation. If the default for the LNKAUTH system parameter is taken (LNKAUTH=LNKLST) or is specified in IEASYSxx or by the operator, libraries in the LNKLST concatenation are authorized when accessed as part of the LNKLST concatenation. If a library is in the LNKLST concatenation, but is not also APF-authorized, referencing this library via a JOBLIB or STEPLIB DD statement causes the library to be considered unauthorized for the duration of the job or step.

Associated Parmlib Member: None

LOGCLS=x

Meaning and Use: This parameter specifies the JES output class for the log data sets. A log data set is queued to this class when its WTL limit has been reached. (The limit is specified by the LOGLMT initialization parameter.)

Example: LOGCLS=L

In this example, the current log data set is queued to output class L when the limit on the number of WTLs has been reached.

If the specified LOGCLS value is invalid, or an I/O error occurs while the IEASYSxx member is being read, master scheduler initialization prompts the operator for a replacement LOGCLS value. If prompting is forbidden (the OPI operand was specified), the default value A is assigned.

For the other log parameter, see LOGLMT.

Value Range: A single alphabetic or numeric character: A-Z or 0-9.

Default Value: A, which represents output class A.

Associated Parmlib Member: None

LOGLMT=nnnnnn

Meaning and Use: This parameter specifies the maximum number of WTLs (messages) allowed for each log data set. The value is used by log processing to determine when a log data set should be scheduled for sysout processing by JES. When the value is reached, log processing issues a simulated WRITELOG command to close and free the current log data set, and to allocate and open a new log data set.

Example: LOGLMT=004852

In this example, when 4,852 WTLs have been issued to a log data set, the data set is scheduled for sysout processing on the output class specified by the

LOGCLS parameter. Log processing then allocates and opens a new log data set.

If the specified value is invalid or an I/O error occurs while the IEASYSxx member is being read, master scheduler initialization prompts the operator for a replacement LOGLMT value. If prompting is forbidden (the OPI operand was specified), the default value of 500 is assigned.

For the other log parameter, see LOGCLS.

Value Range: 000000-999999

Default Value: 500

Associated Parmlib Member: None

LPA= { **aa** | **(aa,bb,...[,L])** }

Meaning and Use: This parameter specifies one or more LPALSTxx parmlib members. The two alphameric characters, represented by aa (or bb, etc.), are appended to LPALST to form the name of the LPALSTxx member(s). If the L option is specified, the system displays (at the operator's console) the names of the data sets successfully concatenated to SYS1.LPALIB.

The LPALSTxx member(s) list data sets that are to be concatenated to SYS1.LPALIB. (For information on the use, contents, and syntax of LPALSTxx, see the description of member LPALSTxx.)

Value Range: Any two alphameric characters.

Default Value: None.

Associated Parmlib Member: LPALSTxx

MAXUSER=nnnnnn

Meaning and Use: This parameter specifies a value that, under most conditions, the system uses to limit the number of jobs and started tasks that can execute concurrently during a given IPL. (The number includes time sharing jobs, batch jobs, started system tasks, the master scheduler, and JES2 or JES3.) However, when the system is heavily used, it can use the value specified on the RSVSTRT system parameter to allow more concurrent jobs and started tasks than the number specified by MAXUSER.

Note: The system uses the values specified for the MAXUSER, RSVSTRT, and RSVNONR system parameters to determine the number of entries in the address space vector table (ASVT), which is used to locate the various address space control blocks.

Assume, for example, that MAXUSER specifies 500 and RSVSTRT specifies 5. If there is an attempt to start an address space (using the START command), and none of the 500 ASVT entries defined by the MAXUSER parameter is available (meaning heavy system use), but an entry defined by the RSVSTRT parameter is available, the system uses that entry. Thus, when the system is heavily used, there can be more concurrent jobs and started tasks in the system than the number defined by MAXUSER.

If the system's recovery has to reconstruct the ASVT because the ASVT was written over in error, some entries reserved for replacement (via RSVNONR) might be added to the normal queue of available entries. Therefore, the absolute limit to the number of concurrent jobs and started tasks is the sum of

the values specified for the MAXUSER, RSVSTRT, and RSVNONR system parameters.

Value Range: 0-32767. Note that the sum of the values specified for the MAXUSER, RSVSTRT, and RSVNONR system parameters cannot exceed 32,767.

Default Value: 255

Associated Parmlib Member: None

MLPA= { **aa** | **(aa[,L][,NOPROT])** | **(aa,bb...[,L][,NOPROT])** }

Meaning and Use: This parameter specifies one or more IEALPAxx parmlib members, which list modules to be added to the pageable LPA, as a temporary LPA extension. Each member lists the names of modules to be loaded from SYS1.SVCLIB, the LNKLST concatenation, and the LPALST concatenation.

The two alphameric characters, represented by aa (or bb, etc.), are appended to IEALPA to form the name of the IEALPAxx member(s). If the L option is specified, the system displays the contents of the IEALPAxx parmlib member(s) at the operator's console as the system processes the member(s).

The LPA modules in the IEALPAxx parmlib member(s) are page protected in storage, by default. If an attempt is made to store into a page-protected module, a protection exception occurs. However, the NOPROT option allows an installation to override the page protection default. When NOPROT is specified, the LPA modules in the IEALPAxx parmlib member(s) are not page protected.

The installation can use the MLPA parameter to temporarily modify an existing LPA at a quick start or a warm start IPL (without creating a new LPA through the CLPA parameter). The added modules are temporary in that they remain as an LPA extension only for the duration of the current IPL. The temporary modules will not be *automatically* reinstated by a quick start or a warm start IPL. That is, the MLPA parameter must be specified again in the next IPL to reinstate the modified LPA.

If the installation wants to retain the temporary modules as a permanent part of the LPA, it should use the IEBCOPY utility or the linkage editor to place the modules in a data set that is part of the LPALST concatenation, and specify the CLPA and LPA parameters at a future IPL to load the specified LPALST concatenation into the LPA.

(For additional information on the MLPA option, see IEALPAxx. For information on the fixed LPA option, see the FIX parameter and the IEAFIXxx member.)

Value Range: Any two alphameric characters, repeated if desired.

Default Value: None. If MLPA is not specified, no modified LPA is created.

Associated Parmlib Member: IEALPAxx

MSTRJCL=xx

Meaning and Use: This parameter specifies the name of the module that contains the JCL used to start the master scheduler address space. Two alphameric characters, represented by xx, are appended to MSTJCL to form the name of a module, MSTJCLxx, which resides in SYS1.LINKLIB.

Using the MSTRJCL system parameter allows an installation to test new master scheduler JCL. For example, test JCL could be stored in MSTJCL01 and selected by using the MSTRJCL parameter (MSTRJCL=01).

MSTRJCL can be specified in the default system parameter list (IEASYS00), IEASYSxx members, or entered by the operator in response to the SPECIFY SYSTEM PARAMETERS message. The default value of 00 selects MSTJCL00 when no value is specified for MSTRJCL or if the operator presses the ENTER key in response to a prompt. MSTJCL00 the csect name in an IBM-supplied module, IEEMSJCL, that contains the JCL used to start the master scheduler address space. SYS1.LINKLIB contains the MSTJCL00. If MSTJCL00 is not found, the operator is prompted until a MSTJCLxx member can be found. The IPL cannot continue without the JCL needed to start the master scheduler.

Value Range: Any two alphameric characters.

Default Value: MSTRJCL=00, causing selection of MSTJCL00.

Associated Parmlib Member: None.

NONVIO= { **dsname** | **(dsname1,dsname2,...,dsnameN)** }

Meaning and Use: This parameter allows an installation to direct VIO paging away from the specified local page data sets. Specify one or more local page data sets that are not to be used for VIO paging as long as space is available on other local page data sets. The page data sets that are designated as non-VIO will contain only address space pages or free slots. However, if space is depleted on the page data sets that allow VIO paging, the non-VIO page data sets will be used for VIO paging. (Using non-VIO page data sets for VIO pages is called an *overflow condition.* The operator is notified if an overflow condition occurs.)

Each dsname specified must be a valid name consisting of a maximum of 44 characters whose format is the same as that required for the PAGE system parameter. Each dsname must specify a data set that was specified as a local page data set on the PAGE=parameter, either in the IEASYSxx parmlib member or by the operator in response to the "SPECIFY SYSTEM PARAMETERS" prompt. If you specify a name that the system cannot identify as the name of a local page data set, then the system ignores that name and issues a message to inform the operator that the data set cannot be recognized as a non-VIO page data set. If you omit the NONVIO parameter, then VIO pages are allowed on all local page data sets.

For cold or quick starts a data set's NONVIO designation is not preserved; you must respecify the NONVIO system parameter. If you do not designate a data set as non-VIO on a cold or quick start, you can designate it in the NONVIO system parameter when you do a subsequent warm start; that data set is marked as non-VIO then, and no more VIO pages will be sent to it (unless an overflow condition occurs). If the non-VIO paging data set remains on the system long enough, all VIO pages on it will eventually migrate, during the normal course of system operation, to other page data sets used for VIO.

For warm starts, a data set's NONVIO designation is preserved. A warm start preserves journaled VIO data set pages. Therefore, all local page data sets that contain VIO pages are required for a warm start. These required data sets would include non-VIO page data sets to which VIO paging was done because of an overflow condition.

Note: During a warm start, a quick start will be forced if (1) a local page data set, not specified as NONVIO, is unavailable or unusable, or (2) a non-VIO

local page data set that contains VIO pages was removed before all of its VIO pages had migrated to other page data sets used for VIO.

If you specify all local page data sets on the NONVIO data set name list, a message is issued to inform the operator of this condition. VIO pages can be written to all local page data sets. Similarly, if the directed VIO function is turned off via the DVIO parameter in the IEAOPTxx parmlib member, all local page data sets can receive VIO pages. If the directed VIO function is turned on again, then auxiliary storage management directs VIO away from any local page data sets designated as NONVIO. The VIO pages on these data sets will eventually migrate to VIO page data sets.

Value Range: Any number of data set names may be specified up to the same limit as exists for the PAGE system parameter.

Default Value: None

Associated Parmlib Member: While there is no directly-associated parmlib member you must specify DVIO=YES in the IEAOPTxx parmlib member in order to activate the directed VIO function; DVIO=NO allows VIO pages to go to any local page data set.

Note: DVIO=YES is the default in the IEAOPTxx parmlib member allowing the use of the directed VIO.

NSYSLX=nnn

Meaning and Use: This parameter allows you to specify the number of slots in the system function table to be reserved for system linkage indexes (LXs). See *SPL: Application Development Guide: Extended Addressability* for information on the system linkage indexes.

If you omit the NSYSLX parameter, the system reserves 55 system linkage indexes. You might need to specify NSYSLX if either of the following conditions is true:

1. Your installation executes applications that request (through the LXRES macro) more than 55 system linkage indexes at a time.
2. An application that owns one or more system linkage indexes fails and is restarted repeatedly. If the application does not reuse its original LX value, the supply is eventually exhausted. This condition requires a re-IPL to reclaim the system linkage indexes.

The total number of system linkage indexes cannot exceed 1024. This total is the sum of the user linkage indexes, plus the system linkage indexes in IEAVXSFM, plus the value specified (up to 512) on the NSYSLX parameter.

If applications use more than 55 system linkage indexes at the same time, specify the NSYSLX value a little higher than the number of system linkage indexes used. If an application that owns one or more system linkage indexes continues to fail, specify the NSYSLX value high enough so that, during execution, enough system linkage indexes are available. This technique means you can put off the re-IPL to reclaim the system linkage indexes, until a time when it is less disruptive.

Value Range: 10-512

Default Value: 55

Associated Parmlib Member: None.

OPI= $\left\{\begin{matrix}\underline{\text{YES}}\\ \text{NO}\end{matrix}\right\}$

Meaning and Use: This parameter specifies whether the operator is to be allowed to override system parameters contained in IEASYSxx members of parmlib. The YES operand allows operator overrides. The NO operand causes overrides to be ignored. If, however, NIP detects an invalid parameter in an IEASYSxx member in which OPI=NO applies, NIP ignores the OPI specification and prompts the operator.

OPI may be specified only in an IEASYSxx member; it may not be specified by the operator.

OPI may be specified either for individual system parameters or for the entire set of parameters.

Examples:

```
IEASYSAA:  MLPA=(00,01),SQA=(10,OPI=NO)
IEASYSBB:  MLPA=(00,01),SQA=10,OPI=NO
```

For IEASYSAA, the operator can override MLPA values but not the SQA value. For IEASYSBB, however, the operator can override neither MLPA nor SQA values.

Note: During system initialization, NIP first uses the IEASYS00 parmlib member to establish parameters, then uses parameters from the operator or any other IEASYSxx parmlib member (identified by the SYSP=xx parameter) to replace already-established parameters or add new ones. The OPI parameter in IEASYS00 carries over to any other IEASYSxx parmlib member identified during the IPL. If you specify OPI=NO in IEASYS00 for an IPL, the parameters affected by the OPI=NO cannot be changed by operator command during the IPL. Even if you use another IEASYSxx parmlib member (via SYSP=xx) and specify OPI=YES for that SYSP=xx, the operator still cannot change any parameters affected by OPI=NO in IEASYS00.

Examples:

In the following IEASYSxx parmlib members, the specification of CSA and SQA in IEASYS00 will prevail for the life of the IPL even if the operator uses SYSP=01 to select IEASYS01 for the initialization process:

```
IEASYS00:  (CSA=100,OPI=NO),MLPA=(00,01),SQA=(10,OPI=NO)
IEASYS01:  (CSA=100,OPI=YES),MLPA=(00,01),SQA=10
```

Value Range: Not applicable

Default Value: YES

Associated Parmlib Member: Not applicable

OPT= $\left\{\begin{matrix}\text{xx}\\ \text{(xx[,L])}\\ \text{(,L)}\end{matrix}\right\}$

Meaning and Use: This parameter specifies a parmlib member that contains the parameters that affect swapping and other decisions made by the system resources manager (SRM). The two alphameric characters, represented by xx, are appended to IEAOPT to form the name of the IEAOPTxx member. If the L option is specified, the system displays the contents of the IEAOPTxx parmlib member at the operator's console as the system processes the member.

If the member cannot be found or contains invalid specifications, SRM prompts the operator to specify an alternate member by respecifying OPT=xx. If the operator cancels the parameter by replying with the ENTER key, SRM uses default values.

The operator can select a new OPT (that is, indicate that the system is to run under the control of an alternate IEAOPTxx member) between IPLs by issuing the SET OPT command.

Value Range: Any two alphameric characters

Default Value: None. If OPT is not specified, SRM uses default values. (See "Part 5: The System Resources Manager" for the default values.)

Associated Parmlib Member: IEAOPTxx

PAGE= { **dsname** | **(dsname1,dsname2,...[,L])** | **(,L)** }

Meaning and Use: This parameter allows the installation to name page data sets as additions to existing page data sets. The maximum number of page data sets is 256, which includes the optional page data set specified by the DUPLEX parameter. The system determines which page data sets to use by merging information from three sources: IEASYS00, IEASYSxx, and the PAGE parameter.

During system initialization, NIP first uses the list of page data sets specified in the PAGE= data set name list of the IEASYS00 parmlib member. It then uses any other IEASYSxx parmlib member (identified via the SYSP=xx parameter). The IEASYSxx PAGE data set name list overrides the one in IEASYS00.

PAGE=dsname and PAGE=(dsname1,dsname2,...[,L]) allow the operator to add page data sets to the list of data sets already specified in IEASYSxx. If the PAGE data set name list in IEASYSxx is null, the operator specification is used.

If the "L" keyword is specified - either in parmlib or from the operator's console - ASM generates a list of all the page and swap data sets that the initialization routines have opened. This list is then displayed at the operator's console.

ASM interprets the final merged sequence of page data set names specified as follows:

- The first named data set on the list is used as the PLPA page data set. This data set contains pageable link pack area (PLPA) pages.
- The second named data set in the list is used as the common page data set. This data set contains all of the common area pages that are not PLPA pages.
- The third and all subsequently named data sets are used as local page data sets. These data sets contain all the system pages (including VIO pages) that are considered neither PLPA nor common data set pages.

Note: To replace local page data sets during an IPL, you must specify the CVIO parameter. (Note that CLPA implies CVIO.)

When defining page data sets, you must ensure that the desired PLPA page data set is the first entry in the data set list, in both IEASYS00 and IEASYSxx. For IEASYS00, this means that the desired PLPA data set name must be in the first sysgen DATASET macro containing the PAGEDSN keyword.

During ASM initialization, there are no checks on the sizes of user-supplied data sets. However, there must be at least three page data sets available for IPL: the PLPA page data set, the common page data set, and at least one local page data set. When ASM initialization completes, the PAGEADD command can be used to add more local page data sets to the system. If the PAGEADD is successful, the temporary page activity reference table (TPARTBLE), which resides on the PLPA page data set, is updated to include those page data sets specified on the PAGEADD command. This updating of the TPARTBLE preserves the most current page data set information so that it can be used for subsequent quick start and warm start IPLs.

The data set intended for PLPA should contain enough space for the entire PLPA, including the extended PLPA. If the entire PLPA cannot fit on this data set, ASM puts the excess on the common page data set. Likewise, if the common page data set gets full, its excess goes to the PLPA page data set. In the interest of good performance, however, you should make the common page data set big enough to prevent its "spilling over" to the PLPA page data set (except in cases forced by error situations). For specific data set size and placement recommendations, see "Part 4: Auxiliary Storage Management Initialization."

How Page Data Sets Are Specified: Page data sets are specified by a merging of information from three sources: 1) IEASYS00 or IEASYSxx; 2) operator-issued PAGE parameter; and 3) the temporary page activity reference table (TPARTBLE). The system merges this information as follows:

- *From the PAGE parameter in IEASYS00:* These data set names were specified in the PAGEDSN keyword of the DATASET macro during system generation.[9] This member is always read although it can have additions or be overridden.
- *From the PAGE parameter in IEASYSxx* (an alternate system list): If the operator selects this list by using the SYSP parameter, the PAGE parameter in IEASYSxx overrides the PAGE parameter in IEASYS00. The overriding parameter can either increase or decrease the number of data sets defined during system generation.
- *From the PAGE parameter specified by the operator in the current IPL:* The system merges this page specification with that in either IEASYS00 or IEASYSxx, but not both. The operator specification of page data sets lasts until the next cold or quick start.
- *From the temporary page activity reference table (TPARTBLE):* The system uses information from the TPARTBLE on quick starts and warm starts (IPLs that do not specify the CLPA parameter). The TPARTBLE contains the names of the page data sets used during the previous IPL, at which time the initialization routines wrote the TPARTBLE to the beginning of the PLPA page data set. If a non-demountable device (such as a 3350) is used for the IPL, the device must be online before the IPL. On subsequent quick starts or warm starts the information in the TPARTBLE is accessible to ASM initialization.

---

[9] For guidance on using the DATASET macro instruction to create page data sets, refer to *System Generation*.

Note: Two other conditions are prerequisite for certain warm start or quick start situations:

1. The local page data sets that contain VIO pages from the previous IPL must be mounted for *all* warm starts, to make VIO slots available. Otherwise, ASM forces a quick start instead.
2. The common page data set from the previous IPL must be mounted for *both* quick starts and warm starts if NIP's writing of the PLPA (and extended PLPA) to the PLPA page data set during the previous cold start resulted in spilling some of the PLPA pages into the common page data set.

In general, page data sets specified by any means must have been allocated, cataloged in the system's master catalog, and preformatted in VSAM format before an IPL can start. You can format the data sets by using the DEFINE PAGESPACE command of access method services (refer to *Access Method Services Reference* for information about the formatting process). If you specify the data sets during sysgen, however, you need not explicitly use the DEFINE processor because the DATASET macro uses DEFINE itself to format the data sets for you.

Syntax Examples for the PAGE Parameter:

Example 1: The following statements each specify one page data set.

```
PAGE=dsname
PAGE=(dsname)
```

Example 2: The following statement specifies three page data sets.

```
PAGE=(dsname1,dsname2,dsname3)
```

Dsname1 holds the PLPA pages, dsname2 holds the common pages, and dsname3 holds the private area pages.

Example 3: The following statement specifies *n* page data sets.

```
PAGE=(dsname1,dsname2,...,dsname<i>n</i>)
```

Dsname1 holds the PLPA pages, dsname2 holds the common pages, and dsname3 through dsname*n* all hold private area pages.

Notes:

1. If the operator specifies the PAGE parameter, ASM initialization adds (but does not replace) the data sets as specified. The PAGE data set name list in IEASYS00 or IEASYSxx contains the first named data sets. To ensure that the operator-specified data sets are used for the PLPA and common page data sets, it is necessary to use an IEASYSxx member that contains a null PAGE parameter; one that does not specify page data sets.
2. It is unnecessary to specify either UNIT or VOLSER because all page data sets must be cataloged in the system's master catalog. ASM initialization therefore does not need externally specified volume serial numbers. The operator may either pre-mount volumes or await a mount message.

Minimum Paging Space: ASM enforces minimum requirements for paging space. If the requirements are not satisfied, ASM is forced to terminate the IPL or, later, the system. Additionally, the use of minimum paging space is inadvisable because it can result in poor performance.

Minimum requirements are as follows:

- There must be at least a PLPA, a common, and a local page data set to IPL the system.
- If no duplex page data set is used (the DUPLEX parameter), the PLPA and common page data sets must be able to hold the total combination of PLPA and common pages (excluding the SQA). As long as no errors occur, the auxiliary storage space for the PLPA and common page data sets is sufficient if the space equals the size of the PLPA and CSA divided between the two data sets.

  If the PLPA and common page data sets spill back and forth because the space is not properly divided between the data sets, performance degradation can result. Severe performance degradation can result if the common page data set is not large enough and, therefore, spills to the PLPA page data set, which is normally read-only after IPL.

  If a duplex page data set is used and is large enough to hold all of the PLPA, including the extended PLPA, and the common pages (the only duplexed areas), the system will be able to continue even if the PLPA and common page data sets are full. Nevertheless, this situation is inadvisable because it defeats the purpose of duplexing and may severely degrade performance.
- Local page data sets are used to hold all private area and those VIO pages not backed by ESTORE. The amount of storage necessary varies with each system and can be calculated using the guidelines in "Part 4: Auxiliary Storage Management Initialization" in this publication.

Page Space Shortage: Two warning messages appear when the system resources manager (SRM) detects a shortage of page space, the first when 70% of the available local paging space has been allocated, and the second when 85% has been allocated. SRM reacts to the situation by preventing the creation of new address spaces. That is, new "start initiator" commands ($SInn), LOGONs, MOUNT commands, and START commands for system tasks that execute in their own address spaces do not work. Upon receipt of these messages, it may be possible to add paging space to the system dynamically by using the PAGEADD operator command. (Refer to *System Commands* for information about using PAGEADD, and to related information about the PAGTOTL parameter in this book.) For these situations, the installation should keep some pre-formatted, cataloged VSAM paging data sets available. The data sets can be formatted by using the DEFINE PAGESPACE processor of access method services (see *Access Method Services Reference)*.

When the page space usage has been decreased below 70% utilization, SRM informs the operator that there is no longer a shortage.

Value Range: Not applicable

Default Value: None

Associated Parmlib Member: None

Note: During NIP processing, the system might exhaust SQA and extended SQA if a large number of local paging data sets was specified on the PAGE parameter. If this condition occurs, the value specified for the SQA parameter may be too low. You can increase the SQA value in IEASYSxx. (See the SQA parameter later in this section.)

PAGTOTL=(ppp,sss)

Meaning and Use: This parameter allows you to specify the total number of page and swap data sets available to the system. ppp includes a plpa, a common, a duplex, and a local page data set as well as page data sets that can be dynamically allocated. This value is valid for the life of the IPL. Space for a duplex page data set is reserved even when duplexing is not active. You also must have at least 1 local page data set to IPL or a WAIT03C will occur. Therefore the minimum number of page data sets needed to IPL is 4.

You can add page and swap data sets by using the PAGEADD operator command until the total number reaches the value specified on the PAGTOTL parameter. However, if you try to exceed the limits set on the PAGTOTL parameter, ASM will truncate the excess specification.

ASM supports a maximum of 256 page data sets (including up to 253 local page data sets, plus PLPA, common, and optional duplex), and 256 swap data sets.

Use this parameter with caution. ASM must reserve SQA space for each page or swap data set that can be dynamically allocated. See the SQA parameter of IEASYSxx.

How Page Number Values Are Obtained: The PAGTOTL parameter is specified in one of two sources: 1) IEASYS00 or IEASYSxx, or 2) by the operator-issued PAGTOTL parameter. An outline of these specifications is as follows:

- *The PAGTOTL parameter in IEASYS00:*
- *The PAGTOTL parameter in IEASYSxx,* an alternate system parameter list: If the operator selects this list (by using the SYSP parameter), the PAGTOTL parameter in IEASYSxx overrides the PAGTOTL parameter in IEASYS00.
- *The PAGTOTL parameter specified by the operator* in the current IPL: This PAGTOTL specification overrides the specification in IEASYS00 or IEASYSxx. The operator specification lasts only for the life of the IPL.
- If the number of page or swap data sets specified on the PAGE= or SWAP= parameters exceeds the PAGTOTL value, the PAGTOTL value will be dynamically increased at IPL. A message is issued if this occurs.

Syntax Example for the PAGTOTL parameter:

```
PAGTOTL=(ppp,sss)
```

This specification causes ASM to allow for the total of of page data sets (*ppp*) and swap data sets (*sss*.)

Value Range: Valid *ppp* values are 0-256. This value is the maximum allowable number of page data sets that may be in use in the paging configuration at any given time. Space for the PLPA, common and duplex page data sets is reserved even when duplexing is not active. Therefore, the maximum number of local page data sets is 253.

Valid *sss* values are 0-256. This value is the maximum number of swap data sets that may be in use in the swapping configuration at any given time.

Default Value: 5,1

Associated Parmlib Member: None

PAK= { **aa** | **(aa,bb,...[,L])** | **(,L)** }

Meaning and Use: This parameter specifies one or more IEAPAKxx parmlib members that contain groups of names of modules in the LPALST concatenation that execute together or in sequence. The two alphameric characters, represented by aa (or bb, etc.), are appended to IEAPAK to form the name of the IEAPAKxx members. If the L option is specified, the system displays the contents of the parmlib member(s) at the operator's console when the system processes the member(s).

Value Range: Any two alphameric characters.

Default Value: PAK=00, causing selection of IEAPAK00, if it exists.

Associated Parmlib Member: IEAPAKxx.

PURGE

Meaning and Use: This parameter specifies that all mass storage system (MSS) volumes currently mounted for this system are to be demounted. PURGE is required when different system configurations have been generated with varying numbers of MSS volumes. If the MSS volumes mounted for this system do not reflect the exact configuration shown in this system's UCBs, some volumes may be inaccessible to the system. Using PURGE to demount all existing volumes forces NIP at I/O initialization to mount the correct volumes for the current system configuration.

Value Range: Not applicable

Default Value: None. (If not specified, no MSS volumes are demounted.)

Associated Parmlib Member: None

RDE= { **YES** | **NO** }

Meaning and Use: This parameter specifies whether the reliability data extractor feature is to be included (YES) or not (NO). For information on RDE, see *SYS1.LOGREC Error Recording*.

Value Range: Not applicable.

Default Value: NO, if you omit RDE or specify an invalid value.

Associated Parmlib Member: IEASYSxx.

REAL=nnnn

Meaning and Use: This parameter specifies the maximum amount of real storage, in 1K blocks, that can be allocated concurrently for ADDRSPC=REAL jobs (that is, V=R jobs). The value is rounded to a multiple of 4K bytes.

Syntax Example: REAL=150 150/4=37.5 pages. Rounding to the next page boundary yields 38 pages, or a value of 152K. This statement allows up to 152K (152 x 1024) bytes to be allocated for use by V=R jobs.

Notes:

1. If possible, avoid a large value for the REAL parameter because a large value degrades system performance even when no REAL regions are allocated.
2. If REAL is specified as zero, no ADDRSPC=REAL job is allowed to run. Furthermore, OLTEP does not run because it requires a REAL region of 76K. Thus, for all practical purposes, 76 is the minimum REAL value.

Value Range: 0-9999. The operand can be from one to four digits. (See Note 2 above.)

Default Value: 76. (This means a default value of 76K.)

Associated Parmlib Member: None

RER= { **YES** / **NO** }

Meaning and Use: This parameter specifies whether the reduced error recovery procedures for magnetic tapes are to be used (YES) or not (NO). This parameter has meaning only when the procedures are stated on the OPTCD parameter on a data definition (DD) statement or on the DCB macro instruction.

Value Range: Not applicable

Default Value: NO, if you omit RER or specify an invalid value.

Associated Parmlib Member: IEASYSxx.

RSU=xx

Meaning and Use: This parameter specifies the number of processor storage units to be made available for storage reconfiguration. The frames in these storage units are not to be used for long-term pages and will be designated the non-preferred area. (Long-term pages include SQA pages, common area fixed pages and LSQA or private area fixed pages associated with non-swappable address spaces.)

An address space is non-swappable:

- If the program name is in the program properties table (PPT) and the appropriate flags are set. (For more information on the PPT, see *System Modifications.*)
- If ADDRSPC=REAL is specified on the JOB or EXEC statement.
- If the address space issues the TRANSWAP sysevent. The DONTSWAP and HOLD sysevents also make an address space non-swappable, but these specifications are considered to be of short duration, and associated LSQA and private area pages are not necessarily put into preferred storage.

During IPL, the operating system assigns the V=R area to the low end of storage and assigns the current SQA to the high end of storage. Then, to satisfy an RSU specification, the system defines reconfigurable storage units, starting with the first offline storage unit at the low end of storage and proceeding upward. If the system cannot satisfy the request from offline storage, it proceeds to define reconfigurable storage units with the online storage units. The system starts with the first online storage unit at the high end of storage and proceeds downward, defining reconfigurable storage units using only those online storage units that do not contain V=R, SQA, LSQA, or nucleus frames.

After the system has defined the reconfigurable storage units, it defines the remainder of the processor storage units as the preferred area for long-term pages.

Notes:

1. The size of a storage unit varies. You can determine the size of the storage units in your processor complex by examining the storage ranges displayed on the configuration frame of the system console. For example, if your storage units are 4 megabytes, the storage ranges displayed on the configuration frame of your system console are 0-3 megabytes, 4-7 megabytes, and so forth. If your storage units are 8 megabytes, the storage ranges displayed are 0-7 megabytes, 8-15 megabytes, and so forth.

   You must determine the storage unit size before you can properly calculate the correct RSU value. For example, assuming a storage unit of 2 megabytes, initializing a 3090 processor with 64 megabytes in physically partitioned mode with RSU=8 causes the operating system to allocate the required 32 megabytes of reconfigurable storage. Specifying an incorrect value on the RSU parameter results in an error message. For more information on specifying RSU values, see *Planning: Recovery and Reconfiguration.*

2. While jobs are processing, short-term pages are assigned to any available processor storage frames in either non-preferred or preferred areas. Long-term pages are assigned only to storage frames in the preferred area. However, when the condition occurs that a long-term page requires storage space but the preferred area is full, the system performs one of the following:

   - *Immediate steal* - If a short-term page is removable (either unchanged or one that can be moved to non-preferred area) and is using a frame in preferred storage, the system removes the page and assigns the new long-term page to the vacated space.
   - *Dynamic expansion* - If all of the frames in the preferred area are being used for pages that are not eligible for "immediate steal," the system looks for a frame in the non-preferred area. When it finds a frame that is not a V=R frame, it converts a portion of the storage unit from non-preferred to preferred.

     Dynamic expansion lowers the number of reconfigurable storage units designated by the RSU parameter. Therefore, the system issues message IAR005I to the system operator. The operator can then issue the display matrix (D M) command to determine which units are still available for reconfiguration.

3. Some system overhead occurs when you specify a value other than zero for the RSU parameter. (RSU=0 is the default.) The overhead results from the system's efforts to keep long-term pages in preferred areas. Generally, the overhead does not cause noticeable performance degradation. However, depending on your installation's workload and requirements for preferred storage, a non-zero RSU specification might cause noticeable performance degradation.

Syntax Example: RSU=4. This example requests that four storage units be made available for storage reconfiguration.

Value Range: 0-99. For processor complexes that cannot be reconfigured, specify RSU=0 (or take the default, which is RSU=0). For processor

complexes that can be reconfigured, specify an RSU value that represents the exact amount of storage needed to reconfigure the processor.

Note: An uncorrectable storage error might prevent you from reconfiguring the processor. If this happens, specify an additional storage unit (on the RSU parameter), at the next IPL, to increase the chance that reconfiguration might work.

If you specify an RSU value that the system cannot fully satisfy, the system defines as many reconfigurable storage units as possible and issues message IAR004I to indicate that the RSU parameter was not completely satisfied. The operator can then issue the display matrix (D M) command to determine how many units were made available for reconfiguration.

Default Value: 0. If the RSU parameter is omitted or specified as 0, all processor storage units are available for preferred storage.

Associated Parmlib Member: None

RSVNONR = nnnnnn

Meaning and Use: This parameter specifies the number of entries in the address space vector table (ASVT) that are to be reserved for replacing entries that are marked non-reusable. A non-reusable entry represents an address space in which a job that terminated had been executing in a cross memory environment. When such a job terminates, the system terminates the address space and marks its associated ASVT entry non-reusable (unavailable) until all of the address spaces the job had cross memory binds with have terminated.

Value Range: 0-32767. Note that the sum of the values specified for the RSVNONR, RSVSTRT, and MAXUSER system parameters cannot exceed 32,767.

Default Value: 5

Associated Parmlib Member: None

RSVSTRT = nnnnnn

Meaning and Use: This parameter specifies the number of entries in the address space vector table (ASVT) that are to be reserved for address spaces created in response to a START command (such as the system management facilities address space and the Library lookaside address space). By reserving entries in the ASVT for such address spaces, you can often avoid having to reinitialize the system because there is no available entry in the ASVT for a critical address space. For example, if LNKLST lookaside (LLA) terminates and you issue a START LLA command to restart LLA, but no ASVT entry is available, LLA will not be restarted. However, if there are reserved entries for critical address spaces (such as LLA), it is more probable that an entry will be available and the LLA address space can be created without having to reinitialize the system.

Value Range: 0-32767. Note that the sum of the values specified for the RSVSTRT, RSVNONR, and MAXUSER system parameters cannot exceed 32,767.

Default Value: 5

Associated Parmlib Member: None

SCH= { **(aa)** | **(aa,L)** | **(aa,bb..)** | **(aa,bb..,L)** }

Meaning and Use: This parameter specifies certain system characteristics. The two alphameric characters, represented by aa,bb.., specify one or more SCHEDxx members of SYS1.PARMLIB. SCHEDxx can include the following statement types:

EDT	Defines which eligible device table the system is to use.
MT	Specifies the size of the master trace table, if one exists.
PPT	Allows an installation to add entries to the program properties table.
RESTART	Enables an installation to add eligible restart codes to the restart codes table.
NORESTART	Enables an installation to delete eligible restart codes from the restart codes table.

If you specify the L option, the system displays the contents of the SCHEDxx SYS1.PARMLIB member at the operator's console as the system processes the member.

If a SCHEDxx member contains invalid specifications, the system prompts the operator to specify another member. The response is in the form SCH=aa. To cancel the parameter, the operator replies by hitting the ENTER key.

For more information on the SCH parameter, see the SCHEDxx member.

Value Range: Any two alphameric characters.

Default Value: None

Associated Parmlib Member: SCHEDxx

SMF=xx

Meaning and Use: This parameter specifies the parmlib member, SMFPRMxx, from which SMF will obtain its options. The two alphameric characters, represented by xx, are appended to SMFPRM to name the member. NIP saves the name until SMF is initialized. (For detailed information on SMF parameters, see member SMFPRMxx.)

Value Range: Any two alphameric characters.

Default Value: 00 (This specifies SMFPRM00, the IBM-supplied default parmlib member.

Associated Parmlib Member: SMFPRMxx

SQA=(a,b)

Meaning and Use: This parameter specifies the sizes of the virtual system queue area (SQA) and extended SQA in multiples of 64K blocks. The subparameter "a" specifies the size of the SQA, located below 16 megabytes. The subparameter "b" specifies the size of the extended SQA, located above 16 megabytes. These values are added to the system's minimum SQA of four 64K blocks (or 256K) and minimum extended SQA of approximately 8 megabytes. Both the SQA and extended SQA are fixed in real storage as they are used.

Notes:

1. During IPL processing, the system reserves four 64K blocks of virtual storage for SQA and fourteen 64K blocks of extended storage for extended SQA. Both of these storage areas are available to fulfill requests for SQA virtual storage during the IPL. This space is the minimum allocation for SQA and extended SQA. It is also called the 'initial allocation of SQA.

   Additionally, the system reserves 8 megabytes of extended SQA to be used for the common area page tables. This storage area is not available to fulfill requests for SQA virtual storage until NIP completes processing the CSA parameter. After the CSA parameter is processed, which means that the size of the common area and the amount of storage required for the common area page tables is known, the reserved storage not needed for the page tables becomes available to fulfill requests for extended SQA virtual storage.

2. During a quick start (that is, when the CLPA parameter is not specified), NIP determines if the currently specified SQA values are equal to the previously specified (cold start) values. If not, NIP issues an informational message and uses the cold-started SQA values. (See the CLPA parameter for a comparison between *cold start* and *quick start*.)

Syntax Example: SQA=(4,5). The first value requests that, for the SQA, (4 x 64K) be added to the system's minimum virtual allocation. The second value requests that, for the extended SQA, (5 x 64K) be added to the system's minimum allocation.

For 3090 processors, MVS determines the amount of extended storage installed on the processor and automatically calculates the blocks of extended SQA necessary to initialize the expanded storage. This amount is added to the minimum system allocation and any increase you specify with the SQA parameter.

SQA Space Shortage: For a GETMAIN storage request for SQA storage where the request can be satisfied with storage either below or above the 16-megabyte line, the virtual storage manager (VSM) will try to satisfy the request with space above 16 megabytes. That is, VSM attempts to satisfy the request with space in the extended SQA, and, if such space is not available, VSM then attempts to satisfy the request with space in the extended common service area (extended CSA). If there is no space available above 16 megabytes, VSM will then try to satisfy the request with space below 16 megabytes, first from the SQA, and finally from the CSA.

For a GETMAIN request for SQA storage where the request must be satisfied with storage below 16 megabytes, VSM tries to satisfy the request with space in the SQA, and, if none is available, VSM then tries to satisfy the request with space in the CSA.

Notes:

1. If excessive amounts of CSA or SQA are allocated, a warning message is generated and the parameter must be respecified. A warning message is also generated when the size of the entire common area below 16 megabytes exceeds 8 megabytes.

   VSM keeps track of the remaining virtual SQA below 16 megabytes, and informs the system resources manager, via a SYSEVENT macro, when the total of available virtual SQA plus available virtual CSA has reached two threshold values. The two values are eight pages, or 32K, and four pages, or 16K. The system resources manager reacts to the situation by issuing an

"SQA shortage" message at each of the two thresholds. At the upper threshold (32K), it also inhibits the creation of new address spaces by disallowing start initiator commands, LOGON commands, MOUNT commands, and START commands for system tasks that require their own address spaces, such as START TCAM.

2. The IPS and the installation control specification are processed before the SQA parameter. If the installation control specification or IPS parmlib member is extremely large, the tables built in the SQA and extended SQA could exceed the initial SQA allocation. Exceeding the allocation can be prevented by increasing the initial allocation within the value range as described below or by putting the IPS and the installation control specification into effect with the SET command after system initialization.

Value Range: On the "a" subparameter, 0-256 (one to three digits) for the SQA. On the "b" subparameter, 0-32511 (one to five digits) for the extended SQA.

Note: During NIP processing, it is possible that the system's minimum allocation for SQA and extended SQA might be depleted before NIP processes the SQA parameter. If this situation occurs, you can increase the minimum SQA and/or extended SQA allocations, using the AMASPZAP service aid to change the contents of NVSQA and/or NVESQA in module IEAIPL04. (See *Service Aids* for information on AMASPZAP). However, if SQA is exhausted after processing local page or swap data sets has begun, the value specified for the SQA parameter may be too low. In this case, do not increase the initial SQA allocation. Instead, increase the SQA value in IEASYSxx.

To increase the initial SQA allocation, change NVSQA and NVESQA, which are both halfwords. NVSQA and NVESQA contain the constants used to define the system's minimum SQA allocation. Increasing the value in either or both fields increases the minimum allocation of SQA and/or extended SQA. For example, to add one 64K block to the SQA available to NIP processing, change the value X'0004' in NVSQA to X'0005'. A similar change to NVESQA adds one 64K block to the minimum extended SQA allocation. If you increase the minimum SQA and/or extended SQA allocations, you should decrease the corresponding value(s) that you specify in the SQA parameter.

See *Data Area* for the locations of NVSQA and NVESQA.

Default Value: For the "a" subparameter, the default is 1. This means one 64K block will be added to the initial SQA allocation of 256K for a default size of 320K for the SQA. For the "b" subparameter, the default is 0. This means no 64K blocks will be added to the minimum extended SQA allocation of approximately 8 megabytes.

Associated Parmlib Member: None

SSN= { **aa** / **(aa,bb,...)** }

Meaning and Use: You specify this parameter to identify the IEFSSNxx parmlib member or members that contain the information needed to define and initialize selected subsystems. The two alphanumeric characters are appended to the characters, IEFSSN, to form the name of the parmlib member. NIP saves the parmlib name or names until the subsystems are to be initialized. (See the description of the IEFSSNxx member in this section for additional information.)

Value Range: Any two alphameric characters

Default Value: SSN=00

Associated Parmlib Member: IEFSSNxx

SVC= { **aa** / **(aa,bb,...)** }

Meaning and Use: The two alphameric characters, (aa, bb, etc.) are appended to IEASVC to form the name of the parmlib member(s). This member contains the installation-defined SVCs. NIP processing uses the member(s) to place the specified SVCs in the SVCTABLE, which resides in the extended read only nucleus.

Value Range: Two alphameric characters

Default Value: None

Associated Parmlib Member: IEASYSxx

SWAP= { **dsname** / **(dsname1,dsname2...)** }

Meaning and Use: This parameter allows the installation to name swap data sets as replacements for existing swap data sets for a maximum of 256 swap data sets in all. This parameter follows the normal override rules for system parameters. In other words, any swap data set specified in a parmlib member IEASYSxx, or specified by the operator, completely overrides the previously specified data sets.

ASM swap data sets are optional. If none are supplied, the pages normally sent to a swap data set are directed to a local page data set. Similarly, if swap data sets become full during system operation, the pages are directed to a local page data set. Nevertheless, it is advisable for sufficient swap space to be available to ASM to prevent possible performance degradation. Additional information and recommendations are in "Part 4: Auxiliary Storage Management Initialization."

How Swap Data Sets Are Specified: Swap data sets are specified from one of two sources: 1) the IEASYS00 or IEASYSxx parmlib member, and 2) the operator-issued SWAP parameter.

- *The SWAP parameter in IEASYS00* - These data set names were specified in the SWAPDSN keyword of the DATASET macro during sysgen.
- *The SWAP parameter in IEASYSxx* (an alternate system parameter list) - If the operator selects this list (by issuing the SYSP parameter), the SWAP parameters in IEASYSxx override the SWAP parameter in IEASYS00. The overriding parameters can either increase or decrease the number of data sets specified during sysgen.
- *The SWAP parameter specified by the operator in the current IPL* - This SWAP parameter specification does not merge with those specified by IEASYSxx, as is the case with the PAGE parameter specification, but instead overrides those specifications. The operator specification lasts only for the life of the IPL.

Before an IPL, SWAP data sets specified through SWAP, or through the DATASET[10] macro at sysgen, must have been allocated, cataloged in the system master catalog, and preformatted in VSAM format, without track overflow. (Unlike some page data sets, swap data sets use multi-tracking instead of track overflow.) You can use the DEFINE PAGESPACE processor of access method

[10] The SWAPDSN keyword of the DATASET macro causes the SWAP parameter to be placed in IEASYS00.

services to format the data sets; refer to *Access Method Services Reference* for information about how to use this processor. For information about using the DATASET macro to create swap data sets, refer to *System Generation*.

Syntax Examples for the SWAP Parameter

Example 1:

SWAP=dsname or
SWAP=(dsname)

Either of these statements specifies one swap data set.

Example 2:

SWAP=(dsname1,dsname2)

This statement specifies two swap data sets.

Example 3:

SWAP=(dsname1,dsname2,...,dsname*n*)

This statement specifies *n* swap data sets.

Note that neither UNIT nor VOLSER is specified. Since all swap data sets must be cataloged, ASM initialization does not need externally specified volume serial numbers. The operator may either pre-mount volumes or await a mount message.

Minimum Swap Space: ASM neither checks for a swap data set nor requires a minimum size if one is specified. If no swap data set is specified, or if swap data set space is exhausted, ASM forces further swap LSQA and working set pages to a local page data set. For this reason, one must consider local page data set sizes.

Swap Space Shortage: If swap data set space becomes unavailable during system processing (as evidenced by message ILR009E, "SWAP DATA SET BAD"), it may be possible to add swap data sets with the PAGEADD operator command. Information about PAGEADD is in *System Commands*. Related information is also available under the "PAGTOTL Parameter" topic and in "Part 4: Auxiliary Storage Management Initialization."

For situations where swap data sets may be added, the installation may keep preformatted, cataloged VSAM swap data sets available.

Value Range: Not applicable

Default Value: None

Associated Parmlib Member: None

SYSNAME=name

Meaning and Use: This parameter specifies the name to be used to identify this system in a global resource serialization complex. The name specified remains in effect for the duration of this IPL. The name must be 1-8 characters long; the valid characters are A-Z, 0-9, $, @, and #. If the name is invalid, the system prompts the operator to respecify it. The default name value is "NONAME." It is a good idea to use the system id (SID) parameter value specified in the SMFPRMxx parmlib member as the name value for SYSNAME.

The system name, defined by the SYSNAME parameter is used during the initialization of global resource serialization to identify the GRSDEF statement in the GRSCNFxx parmlib member that is to apply to this system. The system prompts the operator for a valid system name if the system name specified does

not have the proper syntax. If the system name specified does not match any of the MATCHSYS keyword values in the GRSCNFxx parmlib member, the operator is informed, via an error message, to either re-IPL the system or specify GRS=NONE. The operator also uses the system name defined by SYSNAME parameter to refer to the system in the system commands that control global resource serialization.

SYSNAME will be used in the SYSLOG heading and in all message formats where the system name is requested.

Value Range: 1-8 of these characters: A-Z, 0-9, $, @, #.

Default Value: NONAME

Associated Parmlib Member: None

SYSP= { **aa** / **(aa,bb...)** }

Note: This parameter may be specified only by the *operator*. It cannot validly be specified in a system parameter list. It is included here for the sake of completeness.

Meaning and Use: The SYSP parameter specifies that one or more alternate system parameter lists (e.g., IEASYS01, IEASYS02, etc.) are to be read by NIP in addition to the default list IEASYS00. The two alphameric characters, represented by aa, are appended to IEASYS to name the alternate list(s). Any number of lists are valid. The specification cannot be prohibited by the OPI parameter.

A system parameter value specified in an operator-selected alternate list overrides the value for the same parameter, if it is specified in IEASYS00. (IEASYS00 is always read.) NIP accepts parameters in alternate lists specified in SYSP in priority order from right to left. This means that a parameter defined in a list specified later in SYSP overrides the same parameter defined in a list specified earlier in SYSP.

Example:

The operator responds to SPECIFY SYSTEM PARAMETERS by entering:

```
R 00,SYSP=(01,02)
```

Assume that the two specified members contain the following parameters:

```
IEASYS01:  MLPA=(00,01),SQA=8
IEASYS02:  MLPA=02,SQA=10
```

NIP would accept MLPA=02 and SQA=10, in addition to other parameters contained in IEASYS00.

Value Range: Any two alphameric characters

Default Value: 00 (This specifies IEASYS00.)

Associated Parmlib Member: IEASYSxx

VAL= { **aa** / **(aa,bb,...)** }

Meaning and Use: This parameter specifies the VATLSTxx member or members of parmlib. Volume attribute processing reads this member or members during initialization to obtain *mount* and *use* attributes for direct access volumes. The *mount* and *use* attributes are set in the UCBs whose volume serial numbers are listed in the VATLSTxx member(s). If multiple members are specified, the lists are read in the order in which they appear in the VAL parameter. If a particular volume serial number appears on more than one entry, the volume attributes specified in the last entry for that volume serial will be accepted. If the volume serial does not duplicate another entry, but is for an MSS volume (see the 'device type' parameter in VATLSTxx description), then the last entry with that particular MSS unit address will be used to set the volume attributes for the volume.

(For additional information, refer to the topic that describes the VATLSTxx member.)

Value Range: Any two alphameric characters

Default Value: 00 (This means that VATLST00, if it exists, will be read.)

Associated Parmlib Member: VATLSTxx

VRREGN=nnnn

Meaning and Use: This parameter specifies the default real-storage region size for an ADDRSPC=REAL job step that does not have a REGION parameter in its JCL. The numerical value of the operand (nnnn) indicates the real-storage region size in 1K-byte blocks.

Note: The following VRREGN values will prevent an ADDRSPC=REAL job step from running if it omits a REGION parameter:

- VRREGN value that is greater than the value of the REAL parameter, or
- VRREGN value of zero.

Value Range: 0-9999

Default Value: 64 (This means a default REGION size of 64K.)

Associated Parmlib Member: None

Member Name: IECIOSxx

Use of the Member

IECIOSxx contains records that the installation creates to:

- Modify the time intervals to be used by the missing interrupt handler (MIH) when scanning UCBs for missing interrupts.
- Modify the device threshold values and recovery actions in the hot I/O detection table (HIDT). The system uses the HIDT information to detect, and recover from, hot I/O conditions.

Missing Interrupt Handler (MIH): The MIH periodically scans UCBs to detect missing interrupt conditions. The following conditions qualify as missing interrupts if the specified time interval has elapsed:

- Primary status interrupt pending
- Secondary status interrupt pending
- Start pending condition
- Idle UCB with work queued
- Mount pending

If the MIH detects a missing interrupt, processing will be done that is dependent on the detected condition. For any detected missing interrupt, the MIH performs one or more of the following actions:

- Invokes the device dependent MIH exit (if it exists)
- Records the condition on SYS1.LOGREC
- Notifies the system operator
- Clears the condition
- Simulates an interrupt
- Redrives the device
- Requeues the I/O request

When specifying time intervals, consider the following:

- If the time interval is too short, a false missing interrupt can occur and cause early termination of the channel program. For example, if a 30-second interval is specified for a tape drive, a rewind might not complete before the MIH detects a missing interrupt.
- If the time interval is too long, a job or system could hang because the MIH has not yet detected a missing interrupt. For example, if a 15-minute time interval is specified for a tape drive used as an IMS log tape, the MIH could delay IMS for 15 to 30 minutes because the MIH needs two timer pops to detect the missing interrupt.

Note: To cancel MIH processing for specific devices, specify 00:00 for the time interval value (mm:ss) defined for the associated UCBs.

Hot I/O (HOTIO): "Hot I/O" refers to a hardware malfunction that causes repeated, unsolicited I/O interrupts. If such I/O interrupts are not detected, the system can loop or the system queue area (SQA) can be depleted. IOS tries to detect the hot I/O condition and perform recovery before the system requires a re-IPL.

When the number of repeated, unsolicited I/O interrupts reaches the threshold value defined in the HIDT, IOS assumes that there is a hot I/O condition and proceeds to perform recovery actions. Recovery actions taken for a "hot" device depend on the type of device and its reserve status.

You can use HOTIO statements in IECIOSxx to modify the HIDT, and to eliminate operator intervention where recovery actions defined in the HIDT (by default) require the operator to respond to a message. For additional information on the HIDT, see *System Modifications.*

Note: To cancel hot I/O processing (that is, to have no detection and no recovery from hot I/O conditions) specify 0 for the device threshold value -- DVTHRSH=0.

Parameter in IEASYSxx (or specified by the operator):

IOS=xx

The two alphameric characters (xx) are appended to IECIOS to identify the IECIOSxx parmlib member. If the IOS parameter is not specified, the system uses defaults for MIH processing and defaults for hot I/O processing.

To change the IECIOSxx member after initialization, you can specify the SET IOS=xx command. The SET IOS=xx command changes the MIH intervals; it does not change the values initialized for hot I/O detection. Changing the hot I/O detection values requires a re-IPL.

Syntax Rules

The following rules apply to the creation of IECIOSxx by means of the IEBUPDTE utility:

- Use columns 1 through 71 for data; columns 72 through 80 will be ignored.
- Each record must start with either MIH or HOTIO, followed by one or more blanks.
- The format for each record is:

  MIH parameter[,parameter]. . .
  or
  HOTIO parameter[,parameter]. . .

- If duplicate keywords are found, the last occurrence of the keyword will be used.
- On MIH records, specify the DEV and TIME keywords as a pair. That is, use them together on one record to define time intervals for specific device numbers.
- On MIH records, only one pair of DEV and TIME keywords can appear on any one record. They can appear in any order, and can be separated by other keywords.

- On MIH records, if the same device number exists in more than one DEV keyword, the time interval specified for the last occurrence of the DEV keyword (containing the device number) will be used.

Syntax Examples:

```
HOTIO DVTHRSH=200
MIH STND=02:30,DASD=00:10,DEV=3C0,TIME=01:30
MIH DEV=(2E8-2FF,730-737),TIME=00:30
MIH 3851=15:00
HOTIO DFLT111=(CHPK,CHPF),DFLT112=(CHPK,OPER)
HOTIO DFLT110=(BOX,)
MIH TIME=00:00,DEV=(180-187,230,B10-B17)
```

In the preceding syntax examples, the last record specifies TIME=00:00. This specification cancels MIH processing for the device numbers on the associated DEV keyword. Similarly, you can cancel hot I/O processing by specifying:

```
HOTIO DVTHRSH=0
```

Note: For 3800 devices, the default of 3 minutes for the MIH interval may not be sufficient. The recommended MIH interval for a 3800 is 5 minutes.

IBM-Supplied Default

There is no default IECIOSxx parmlib member. However, there are default time intervals for MIH processing, and a default device threshold value, as well as default recovery actions, for hot I/O processing. (The following parameter descriptions define the defaults.)

Internal Parameters for MIH

Parameter	Meaning and Use	Value Range	Default
CHAR = mm:ss	Specifies the time interval to be used for the character reader device class, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	03:00
COMM = mm:ss	Specifies the time interval to be used for the communications device class, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	03:00
CTC = mm:ss	Specifies the time interval to be used for the channel-to-channel device class, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	03:00
DASD = mm:ss	Specifies the time interval to be used for all direct access devices, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	00:15
DEV = devnum	Specifies the device number(s) for which specific time intervals are to be used. To specify more than one device number, use a comma to separate the device numbers. To specify a range of device numbers, use a hyphen to separate the beginning and ending device numbers. If more than one device number is specified, enclose the device numbers in parentheses. **Note:** The DEV keyword must precede or follow the TIME keyword on the same record.	000-FFF (hex) for each device	None
GRAF = mm:ss	Specifies the time interval to be used for the graphics reader device class, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	03:00
HALT = mm:ss	Specifies the time interval to be used for monitoring Halt (HSCH) and Clear(CSCH) I/O instructions.	mm = 00-99 ss = 00-59	00:05
MNTS = mm:ss	Specifies the time interval to be used for monitoring 'mount pending' conditions for DASD and TAPE devices.	mm = 00-99 ss = 00-59	03:00
MOUNTMSG = YES|NO	Specifies that MIH is to issue all MIH mount pending messages.		NO
STND = mm:ss	Specifies the time interval to be used for all UCBs, for unit records, tapes, graphics, communications, CTCs, and character readers, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	03:00
TAPE = mm:ss	Specifies the time interval to be used for the tape device class, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	03:00
TEST	Specifies the system is to dynamically test MIH parameters and values. The TEST parameter is ignored during initialization. The system uses this parameter when you issue a SET IOS = xx command to syntax check the MIH updating.		
TIME = mm:ss	Specifies the time interval to be used for the devices identified on the DEV keyword, where mm is minutes and ss is seconds. **Note:** The TIME keyword must precede or follow the DEV keyword on the same record.	mm = 00-99 ss = 00-59	None
UREC = mm:ss	Specifies the time interval to be used for the unit record device class, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	03:00
VDAS = mm:ss	Specifies the time interval to be used for MSS virtual DASD, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	12:00
3851 = mm:ss	Specifies the time interval to be used for the Mass Storage Controller, where mm is minutes and ss is seconds.	mm = 00-99 ss = 00-59	12:00

Internal Parameters for HOTIO

Parameter	Meaning and Use	Value Range	Default
DVTHRSH=ddddd	Specifies the device threshold for unsolicited I/O interrupts, which must be reached for hot I/O recovery to begin.	0-32767	100
DFLT110=(options)	Specifies the recovery action to be taken for a non-DASD, non-dynamic pathing device. (For the recovery actions that can be specified as options, see "Options for HOTIO Recovery.")		(BOX)
DFLT111=(options)	Specifies the recovery action to be taken for a DASD or a dynamic pathing device that is not reserved. (For the recovery actions that can be specified as options, see "Options for HOTIO Recovery.")		(CHPK,BOX)
DFLT112=(options)	Specifies the recovery action to be taken for a DASD or a dynamic pathing device that is reserved. (For the recovery actions that can be specified as options, see "Options for HOTIO Recovery.")		(CHPK,OPER)

Options for HOTIO Recovery: The parameters (DFLT110, DFLT111, and DFLT112) that are used to specify recovery actions have default options defined in the HIDT. Each of these parameters has a default option for non-recursive hot I/O conditions and a default option for recursive hot I/O conditions. You can override one or both options.

For example, if you want to override the DFLT111 non-recursive default option (CHPK) so that the recovery action would have to be obtained from the operator, code: DFLT111=(OPER,). Note that the options are enclosed in parentheses and separated by a comma.

The following options can be specified for both non-recursive and recursive hot I/O conditions. Code the options with the DFLT110, DFLT111, and DFLT112 parameters, and use the following format:

```
Parameter=(a,b)
```

where:

- **a** is the action to be taken for a non-recursive hot I/O condition.
- **b** is the action to be taken for a recursive hot I/O condition.

Option	Meaning and Use
BOX	Specifies that the device is to be forced offline.
CHPF	Specifies that the channel path over which the last interrupt for the hot device was received is to be forced offline.
CHPK	Specifies that channel path recovery is to be initiated for the channel path over which the last interrupt for the hot device was received. If successful, the channel path is to remain online.
OPER	Specifies that the recovery action is to be obtained from the operator.

Member Name: IEFSSNxx

Use of Member

IEFSSNxx is a parmlib member that contains parameters defining the primary subsystem and the various secondary subsystems that are to be initialized during system initialization. IEFSSNxx allows you to name the subsystem initialization routine to be given control during master scheduler initialization. IEFSSNxx also allows you to specify the input parameter to be passed to the subsystem initialization routine. Using the PRIMARY keyword, you can specify a primary subsystem name. The NOSTART keyword used with the PRIMARY keyword indicates that the system is not to issue an automatic start for the primary subsystem.

The order in which the subsystems are initialized depends on the order in which they are defined in the IEFSSNxx parmlib member on the SSN parameter. Initialize the primary (JES) subsystem first. Some subsystems require the services of the primary subsystem in their initialization routines. Problems can occur if subsystems that use the subsystem affinity service in their initialization routines are initialized before the primary subsystem.

The format of the IEFSSNxx record for the Storage Management Subsystem (SMS) is included in the description of the IGDSMSxx parmlib member later in this chapter. If you are activating the SMS, specify its record before you specify the primary subsystem record.

The SSN parameter in IEASYSxx identifies the IEFSSNxx member that the system is to use to initialize the subsystems.

Parameter in IEASYSxx (or specified by the operator):

SSN= { **aa** / **(aa,bb,...)** }

The two-character identifier, represented by aa (or bb, and so forth) is appended to IEFSSN to identify IEFSSNxx members of parmlib. If the SSN parameter is not specified, the system uses the IEFSSN00 parmlib member.

The order in which the subsystems are defined on the SSN parameter is the order in which they are initialized. For example, a specification of SSN=(13,Z5) would cause those subsystems defined in the IEFSSN13 parmlib member to be initialized first, followed by those subsystems defined in the IEFSSNZ5 parmlib member.

Syntax Rules

The following rules apply to the creation of IEFSSNxx by means of the IEBUPDTE utility.

- Each record in IEFSSNxx defines one and only one subsystem that is to be initialized.
- Parameters begin with the comma following the *init-routine* and end with the first blank or comma.
- Each record in IEFSSNxx is 80 bytes long and has the following format:

---

ssname[,init-routine[,parm]][,PRIMARY[,NOSTART]] comments

---

ssname
The subsystem name. The name can be up to 4 characters long; it must begin with an alphabetic or national character, and the remaining characters (if any) can be alphanumeric or national. The name begins with the first non-blank character in the record and continues to the first comma or blank. When a blank follows the name, the system assumes that no initialization routine exists and that only comments (if any) follow.

init-routine
The name of the subsystem initialization routine. This name can be 1-8 characters long, and the characters can be alphanumeric or national. The name begins with the first character following the first comma after "ssname" and continues to the next comma or blank. When a blank follows the name, the system assumes that no initialization routine input parameters are supplied and that only comments (if any) follow.

parm
Input parameters to be passed to the subsystem initialization routine. The input parameters are variable in length for the remainder of the 80-byte record (for a maximum of 60 characters); they begin after the comma that ended the "init-routine" name and end with the first blank. If blanks, commas, or apostrophes are included in the input parameters, then the entire parm field must be enclosed in apostrophes. If the parm field is enclosed in apostrophes, an apostrophe within the field must be specified as a double apostrophe.

PRIMARY
Parameter indicating the primary subsystem name. This keyword indicates that the subsystem name (ssname) will be the primary subsystem name. Initialize the primary subsystem before any secondary subsystem(s). If you specify PRIMARY more than once, the system issues a message.

NOSTART
Parameter indicating that an automatic start is not to be issued. NOSTART indicates that an automatic start of the primary subsystem is not to be issued. NOSTART cannot be used without the PRIMARY parameter.

comments
Comments can be specified after the first blank in the record following the input parameters, or PRIMARY and NOSTART, for as many characters as remain in the 80-byte record.

You can include any number of records in the IEFSSNxx parmlib member.

IBM-Supplied Defaults

If you do not specify the the SSN system parameter, the system uses the IEFSSN00 parmlib member. JES2 is the default for the primary job entry subsystem.

Internal Parameters

None

Member Name: IGDSMSxx

Use of the Member

IGDSMSxx contains the parameters that initialize the Storage Management Subsystem (SMS) and define it to the system.

Use the SET SMS command to select or replace the IGDSMSxx member that SMS is to use. Use the SETSMS command to:

- Activate an SMS configuration
- Change the ACDS that SMS is using
- Change the communications data set
- Change the synchronization interval
- Specify trace options for SMS

For complete descriptions of SETSMS and SET SMS, see *System Commands.*

IGDSMSxx is identified to the system by a record in IEFSSNxx; the record is described in the following section.

Format for the IEFSSNyy Record

To initialize SMS during IPL, place a record in the IEFSSNxx member to identify the IGDSMSxx member. This record differs somewhat from the one described under IEFSSNxx member in this chapter; do not use the optional parameters PRIMARY or NOSTART. The format of the SMS record in IEFSSNxx is:

```
SMS[,IGDSSIIN][,'ID=xx][,PROMPT={YES      }]'
                                {DISPLAY  }
                                {NO       }
```

The fields within the SMS record are as follows:

SMS
Identifies the subsystem as SMS.

IDGSSIIN
Identifies the SMS subsystem initialization routine.

ID=*xx*
Identifies the IGDSMSxx member that contains the SMS options. If you do not code ID, the system uses IGDSMS00. Enclose the ID and/or PROMPT parameters in apostrophes.

PROMPT
Indicates the amount of control the operator has over the rest of the SMS initialization parameters. Enclose the ID and/or PROMPT parameters in apostrophes.

YES
Requests that the system issue a write-to-operator with reply (WTOR) to prompt the operator to change the parameters specified in the IGDSMSxx member. The system displays the current status of parameters before it issues the WTOR. The effect of any change lasts only for the duration of the IPL; the IGDSMSxx member does not change.

DISPLAY
Requests that the system display the contents of the IGDSMSxx member. The operator cannot, however, change the parameters.

NO
Requests that the system is not to display the parameters and the operator is not to change them. **PROMPT=NO** is the default.

Example: The following record defines SMS to MVS IGDSMSST is the IGDSMSxx member that contains the SMS options, and SMS is to prompt the operator for changes to the SMS parameter options.

```
SMS,IGDSSIIN,'ID=ST,PROMPT=YES'
```

Syntax Rules for IGDSMSxx

You can separate the keywords in IGDSMSxx by blanks or commas. You do not need continuation characters for multiple lines.

IBM-Supplied Defaults

The default member is IGDSMS00. The user must supply this member.

Internal Parameters

Parameter	Meaning and Use	Initial Values
SMS	Specifies that this record is for SMS. You must code this parameter.	
ACDS(acds)	Specifies the name of the active control data set (ACDS). If you do not specify this parameter, SMS asks the operator to supply it. For information about the SMS data sets and routines, see *Storage Administration Reference*.	
COMMDS(commds)	Specifies the name of the communications data set that SMS uses. If you do not specify this parameter, SMS asks the operator to supply it.	
DINTERVAL(seconds)	Specifies the amount of time (1 to 999) that SMS is to wait between reading the device statistics.	150 seconds
INTERVAL(seconds)	Specifies the amount of time (1 to 999) that SMS is to wait before synchronizing with any other SMS systems in the complex.	15 seconds
REVERIFY(YES|*NO*)	Directs SMS to check a user's authority to allocate a new data set and use storage or management class at job interpretation time or at both job interpretation time and execution time. If you want SMS to check the authority at both times, code YES; NO directs SMS to check only at job interpretation time.	NO
ACSDEFAULTS(YES|*NO*)	Specifies how SMS is to get values for the automatic class selection (ACS) routine variables. Specify YES to use ACS routine defaults. Specify NO to request that RACF or a functional equivalent give SMS the values. Because SMS must set these variables every time a data set is created, specifying YES reduces the overhead of using RACF. The ACS routine variables are: &DSOWNER &APPLIC &DEF_DATACLAS &DEF_MGMTCLAS &DEF_STORCLAS	NO
TRACE(*ON*|OFF)	Specifies whether tracing is to be on (ON) or off (OFF).	ON
SIZE(nnnnnnK|nnnM|nnnnnn)	Specifies the size of the trace table. *nnnnnnK* specifies the size in kilobytes (rounded up to the nearest 4K); the maximum is 255,000 kilobytes. *nnnM* specifies the size in megabytes; the maximum is 255 megabytes. If you specify *nnnnnn* without a unit, the system assumes a unit of kilobytes. The minimum is 0.	128K
TYPE(*ERROR*|ALL)	Specifies whether SMS is to trace an error entry (ERROR) or all entries (ALL).	ERROR
JOBNAME(jobname|*)	Specifies whether SMS is to trace a specific job (*jobname*) or all jobs (*).	*
ASID(asid|*)	Specifies whether SMS is to trace a specific address space (*asid*) or all address spaces (*).	*

Parameter	Meaning and Use		Initial Values
SELECT(*ALL*[,option...])	Specifies one or more events or services that SMS is to trace. The options are:		ALL
	MODULE	Module entry or exit	
	SMSSJF	Storage Management Subsystem/SJF interfaces	
	SMSSSI	Storage Management Subsystem/SSI interfaces	
	ACSINT	ACS services interfaces	
	OPCMD	Operator commands	
	CONFC	Configuration changes	
	CDSC	Control data set changes	
	CONFS	Configuration services	
	MSG	Message services	
	ERR	Error recovery and recording services	
	CONFR	Return data from an active configuration	
	CONFA	Activate a new configuration	
	DCF	Trace SMS read statistics, Cache maintenance and attribute selection	
	ACSPRO	Perform ACS processing	
	IDAX	SMS interpreter/dynamic allocation	
	DISP	DISP processing exit	
	CATG	SMS catalog services	
	VOLREF	SMS VOLREF services	
	SCHEDP	Scheduling services (pre-locate catalog orientation)	
	SCHEDS	Scheduling services (system-select)	
	VTOCL	VTOC/data set services (allocate existing data set)	
	VTOCD	VTOC/data set services (delete existing data set)	
	VTOCR	VTOC/data set services (rename existing data set)	
	VTOCC	VTOC/data set services (create new data set)	
	VTOCA	VTOC/data set services (add a volume to a data set)	
	RCD	Recording services	
	ALL	All of the above options	
DESELECT(option[,option...])	Delete one or more events from the list of events to be traced. The events are the same as those listed under the SELECT parameter.		

Example of Contents of IGDSMSxx

You want to define SMS with the following properties:

- An ACDS of SYS1.ACDS9
- A communications data set of SYS1.COMMDS
- An interval of 10 seconds before synchronizing with any other SMS subsystems in the complex
- Tracing of all events, except operator commands, message service, and DISP processing exits

To request that SMS have these properties, code the following statement in IGDSMSxx member of parmlib:

```
SMS ACDS(SYS1.ACDS9) COMMDS(SYS1.COMMDS) INTERVAL(10)
    TRACE(ON) TYPE(ALL) DESELECT(OPCMD,MSG,DISP)
```

Member Name: IKJPRM00

Use of the Member

IKJPRM00 is an optional member that contains installation-defined TIOC parameters used mainly to control TSO/TCAM time sharing buffers. IKJPRM00 is used only during TIOC initialization and does not participate in system initialization.

If the installation uses TSO/TCAM time sharing, the system programmer may optionally construct this member using the IEBUPDTE utility. (See example of IEBUPDTE statements in chapter introduction.) The default values, listed under "Internal Parameters" later in this section, are internal constants of the TIOC program. You may override a default value by placing the same parameter into the member.

IKJPRM00, or an alternate member name, may be specified by the operator as an optional parameter of the MODIFY tcamproc command. The command starts TSO/TCAM time sharing under MVS. The command syntax consists of:

MODIFY tcamproc,TS=START [,member name]

Member name can be either defaulted to IKJPRM00, or specified as the name of an installation-defined alternate. If the operator omits the member name, the system looks for member IKJPRM00 when time sharing is started. (For additional information on the use of the MODIFY tcamproc command, see *System Commands.*)

TIOC initialization tries to obtain parameters by reading the specified parmlib member. Special processing occurs if errors are encountered. If parmlib can't be allocated or opened, an information message is issued, and default parameters are used.[11] If the specified member can't be found in parmlib, another message is issued and TIOC initialization terminates. In this case, the operator should reenter the MODIFY tcamproc command, either specifying the correct member name or omitting the member name. If the name is omitted, TIOC initialization tries to read IKJPRM00. If it can't locate IKJPRM00, or encounters an I/O error in reading the explicit or default member, it uses the default parameters. If TIOC initialization encounters an invalid parameter, which is not correctly specified in a later entry, it uses the default value. Unsupported parameters, if retained from a previous version of IKJPRM00, are ignored.

Parameter in IEASYSxx (or specified by the operator):

None

[11] See "IBM Supplied Defaults" later in this section.

Syntax Rules

The following rules apply to the creation of IKJPRM00 by means of the IEBUPDTE utility:

- Each record must start with the word TIOC, followed by a blank.
- For each record, columns 1 through 71 are valid for data. Columns 72 through 80 are ignored.
- A parameter must be complete in a record. It may not cross record boundaries. The parameter, however, may be repeated.
- When a parameter is specified more than once in the member, the last occurrence is accepted.
- You may use either a blank or a comma as a separator between adjacent keywords.
- Invalid or misspelled parameters are ignored. Defaults are substituted, and an informational message is issued to the operator.

IBM-Supplied Defaults

The default values are:

BUFSIZE=64,BUFFERS=6xUSERMAX,USERMAX=number of time-sharing terminals[12] + 10%,OWAITHI=20,OWAITLO=4, INLOCKHI=4,INLOCKLO=1,RESVBUF=BUFFERS/10,RECONLIM=0

[12] The number of time sharing terminals is the number of TCAM terminals defined as usable for time-sharing. (See *TCAM Installation Reference* for information on defining terminals for time-sharing.)

Internal Parameters

Parameter	Meaning and Use	Value Range	Default
BUFSIZE	Specifies the storage size of a TIOC buffer.	20-252	64
BUFFERS	Specifies the number of buffers in the TIOC buffer pool. (See note under the OWAITHI parameter.)	4-32,767	six times the USERMAX value
INLOCKHI	Specifies the number of TIOC buffers to be allocated to a terminal user for input before his keyboard is locked. This is not an exact lock but works on an input line basis. If the number of buffers used to input one or more lines exceeds the INLOCKHI value, the keyboard remains locked until all of these conditions are satisfied: the user is swapped in, part or all of the input is removed (the TGET is satisfied), and the number of allocated buffers is reduced to or below the INLOCKLO value. INLOCKHI must be large enough to permit TIOC to receive the largest possible legitimate input message sent from any terminal in the system. Note that when using the FIELD MARK key on 3270 terminals to enter a chain of commands, at least one TIOC buffer is required for each command in the chain. Any input message larger than BUFSIZE times INLOCKHI (or any chain of commands exceeding the number of available TIOC buffers) will be canceled and will cause an error message at the terminal.	1-253	4
INLOCKLO	Specifies a low threshold of allocated input buffers When the number of allocated input buffers is reduced to or below this number, the user's keyboard is unlocked.	less than INLOCKHI and BUFFERS	1
OWAITHI	Specifies the maximum number of output buffers that can be allocated to a terminal. When that number is reached, the user's address space is placed in output wait and is swapped out of real storage. **Note:** If your installation uses the 3270 terminal, specify enough buffers to completely fill the screen. You may compute this number of buffers from the formula: $\text{Buffers} = \frac{(\text{message length}+6)}{(\text{BUFSIZE}-12)}$ If there are not enough buffers for a "full screen write," the address space will be put into output wait and swapped out until buffers become available.	1-253	20
OWAITLO	Specifies a low threshold value for the number of allocated output buffers. When the number of output buffers reaches this value, the system resource manager is notified that the terminal user's job can be swapped into storage and allowed to execute.	less than OWAITHI and BUFFERS	4

Parameter	Meaning and Use	Value Range	Default
RECONLIM	Specifies the time limit in minutes within which a user may reconnect after his TP line has been disconnected.	0-32,767	0
RESVBUF	Specifies the minimum number of free buffers that are available. Its purpose is to maintain a reserve of free buffers that can handle output without "bottlenecking" the system. If the number of free buffers falls below this value, all terminals are locked for input, regardless of INLOCKHI value. The terminals will be unlocked when the number of free buffers becomes equal to RESVBUF.	1-value of BUFFERS	10% of the number of buffers specified in BUFFERS parameter.
USERMAX	Specifies the maximum number of time-sharing users that may be logged on.	1-32,767	Total number of terminals that support time sharing +10%. (**Note:** The number of terminals that support time sharing is specified in the TCAM/TSO message handler, for more information see "Time-Sharing Support" in *TCAM Installation Reference*

Member Name: IKJTSO00

Use of the Member

As of TSO Extensions (TSO/E) Version 1 Release 4, IKJTSO00 is available to identify the commands and programs the system is to use. This member allows you to identify the authorized commands and programs, the commands that a user cannot issue in the background, and the APF-authorized programs that users may call through the TSO service facility. It also allows you to specify the defaults for the TSO/E SEND command. By defining certain options on the SEND parameter in IKJTSO00, you can specify the data set that the SEND command is to store messages in, this is the data set that the LISTBC processing will check when retrieving stored messages.

To use IKJTSO00, copy the default from SYS1.SAMPLIB(IKJTSO00) to SYS1.PARMLIB(IKJTSO00). You can then update the member to meet the needs of your installation.

If your installation is using the virtual lookaside facility (VLF), be sure to add VLFNOTE to this parmlib member.

Parameter in IEASYSxx (or specified by the operator):

None

Syntax Rules

1. Delimit comments with a '/*' and '*/'.
2. Use a plus sign (+) or a dash (-) to continue a line.

IBM-Supplied Defaults

None

Internal Parameters

Parameter	Meaning and Use
AUTHCMD NAME(*cmd1,cmd2...*)	Specifies the authorized TSO commands. ***cmd1,cmd2*** list the authorized commands. Each can contain up to eight characters. See SYS1.SAMPLIB for the current list of authorized commands.
AUTHPGM NAME(*pgm1,pgm2...*)	Specifies the authorized programs. ***pgm1,pgm2*** list the authorized programs. Each program name can contain up to eight characters. SAMPLIB contains the following programs: Utilities RACF IEBCOPY ICHUT100 ICHUT200 ICHUT400 ICHUEX00
AUTHTSF NAME(*name1,name2...*)	Specifies the APF-authorized programs that may be called through the TSO service facility. ***name1,name2*** identify the names of the programs. Each name can contain up to eight characters. SAMPLIB contains the following programs: IEBCOPY IKJEFF76
NOTBKGND NAME(*cmd1,cmd2...*)	Specifies the commands that may **not** be issued in the background. ***cmd1,cmd2*** list these commands. Each can contain up to eight characters. SAMPLIB contains the following commands: OPERATOR OPER TERMINAL TERM
SEND	Specifies the installation's defaults for the TSO/E SEND command. The defaults are shown here.
OPERSEND(ON/OFF),	OPERSEND specifies whether or not users authorized to use the OPERATOR command can issue the SEND subcommand to send messages or notes. See *TSO/E Commands*
USERSEND(ON/OFF),	USERSEND specifies whether or not users can issue the SEND command to send messages or notes to other terminal users.
SAVE(ON/OFF),	SAVE specifies whether or not the SEND command processor and the OPERATOR SEND command processor are to save messages in a log that the installation specifies.
CHKBROD(ON/OFF)	CHKBROD indicates whether LISTBC processing is to check both the SYS1.BRODCAST and the installation's log for messages. ***ON*** indicates that LISTBC is to check both SYS1.BRODCAST and the installation-specified log. ***OFF*** indicates that LISTBC processing is to check only the installation-specified log for saved messages.
LOGNAME (DSNAME/SYS1.BRODCAST)	LOGNAME identifies the log name of the data set where the system saves messages and notes. ***DSNAME*** identifies one or more lower level qualifiers of the installation-specified log. The system automatically sets the first level qualifier to the user's ID. Each qualifier in ***DSNAME*** can be one to eight characters. For more information on setting the installation's defaults with the SEND parameter, see *TSO/E Customization*

Member Name: IKJTSOxx

Use of the Member

As of TSO Extensions (TSO/E) Version 2 Release 1, IKJTSOxx is available to identify the commands and programs the system is to use. This member allows you to identify the authorized commands and programs, the commands that a user cannot issue in the background, and the APF-authorized programs that users may call through the TSO service facility. It also allows you to specify the defaults for the TSO/E ALLOCATE, SEND, RECEIVE, TRANSMIT, and TEST commands. By defining the SHR option on the ALLOCATE parameter, you can change the system default for the disposition of data sets from OLD to share (SHR). The TRANSREC parameter allows you to specify the characteristics for the data to be transmitted or received. By defining certain options on the SEND parameter in IKJTSOxx, you can specify the data set that the SEND command is to store messages in, and the data set that the LISTBC processing is to check when retrieving stored messages. The TEST parameter allows you to specify the installation-written test subcommands and the names of TSO commands that are to be allowed to execute under the TEST command. These commands are in addition to those allowed by default.

To use IKJTSOxx, copy the default from SYS1.SAMPLIB(IKJTSO00) to SYS1.PARMLIB(IKJTSOxx). You can then update the member to meet the needs of your installation.

If your installation is accessing the virtual lookaside facility (VLF) be sure to add VLFNOTE to this parmlib member.

Notes:

You can use IKJTSOxx under TSO/E Version 1 Release 4 with MVS/SP Version 3. However, the following parameters are not supported in IKJTSO00:

- ALLOCATE
- RECEIVE
- TRANSMIT
- TEST

Parameter in IEASYSxx (or specified by the operator):

None

Selecting the IKJTSOxx member

If present, IKJTSO00 is used automatically during IPL. To select another IKJTSOxx member after IPL, you can issue the following TSO command:

```
PARMLIB  UPDATE(xx)
```

Where xx is the alphameric value to be appended to IKJTSO.

To list the values in the active IKJTSOxx parmlib member, you can issue the following command:

```
PARMLIB  LIST(ALLOCATE)
              (AUTHCMD)
              (AUTHPGM)
              (AUTHSF)
              (NOTBKGND)
              (SEND)
              (TEST)
              (TRANSREC)
              (ALL)
```

See *TSO/E Version 2 System Programming Commands Reference* for more information about the PARMLIB command.

Syntax Rules

1. Delimit comments with a '/*' and '*/'.
2. Use a plus sign (+) or a dash (-) to continue a line.

IBM-Supplied Defaults

IKJTSO00 is supplied and used automatically be used during IPL. To use another IKJTSOxx member, issue the PARLIB UPDATE(xx) command after IPL.

Internal Parameters

Parameter	Meaning and Use
ALLOCATE DEFAULT { (OLD) / (SHR) }	Allows you to specify the default value for the data sets required by a program. If you do not specify the ALLOCATE parameter, the system defaults to OLD.
AUTHCMD NAME(*cmd1,cmd2...*)	Specifies the authorized TSO commands. ***cmd1,cmd2*** list the authorized commands. Each can contain up to eight characters. See SYS1.SAMPLIB for the current list of authorized commands. If you are using VLF, make sure that the VLF command (VLFNOTE) is included in the IKJTSOxx member you are using.
AUTHPGM NAME(*pgm1,pgm2...*)	Specifies the authorized programs. ***pgm1,pgm2*** list the authorized programs. Each program name can contain up to eight characters. SAMPLIB contains the following programs: **Utilities RACF** IEBCOPY ICHUT100 ICHUT200 ICHUT400 ICHUEX00
AUTHTSF NAME(*name1,name2...*)	Specifies the APF-authorized programs that may be called through the TSO service facility. ***name1,name2*** identify the names of the programs. Each name can contain up to eight characters. SAMPLIB contains the following programs: IEBCOPY IKJEFF76
NOTBKGND NAME(*cmd1,cmd2...*)	Specifies the commands that may **not** be issued in the background. ***cmd1,cmd2*** list these commands. Each can contain up to eight characters. SAMPLIB contains the following commands: OPERATOR OPER TERMINAL TERM
TEST TSOCMD(*cmd1,cmd2,cmd3.....*) SUBCMD((*scmd1,load1*), (*scmd2,load2*)...)	The TEST parameter specifies that the following commands are authorized to be executed in a test environment. TSOCMD specifies that the following commands (***cmd1,cmd2...***) are installation-written TSO commands that are allowed to be executed under TEST. SUBCMD specifies that the following installation-written command can be invoked as a subcommand of TEST. The value for ***scmd1*** is the command. The value for ***load1*** is the entry point for the program to be invoked as the subcommand. For each SUBCMD specified, you must include both the command and the program name for the command.
TRANSREC	TRANSREC allows you to specify the characteristics for the RECEIVE and TRANSMIT commands. See *SPL: TSO Commands and Macros* for the complete descriptions of the keywords.
NODESMF (*nodename1,smfid1*),(*nodename2,smfid2*),...	The NODESMF parameter builds a table specifying the correspondence between the system identifiers and the network node names. ***nodename*** specifies the name of the network node. The value specified here must be the name of a node defined on the NJERMT JES2 initialization statement or on the JES2 initialization parameter defining the node (DEFINE NODE, Nnnnn). ***smfid*** specifies the system identifier for a particular processor. The value specified here must be the same as a system identifier defined in the SMFPRMxx parmlib member on the SID parameter.
SPOOLCL(*spoolclass*)	
CIPHER [(***ALWAYS***) / (YES) / (***NO***)]	CIPHER indicates the installation specification for controlling data encryption. ***ALWAYS*** indicates that for every transmission, the data will be automatically encrypted. ***YES*** indicates that encryptions is a user option. ***YES*** is the default. ***NO*** indicates that encryptions is not allowed on any transmission.

Parameter	Meaning and Use
OUTWARN(*n1,n2*)	Allows the installation to specify a warning to a user who is transmitting a large file. ***n1.*** specifies the number of records to be transmitted before a warning message is issued. ***n1.*** is a decimal and the default is 10000. ***n2.*** specifies the number of records to be transmitted before a second warning message is issued. ***n2.*** is a decimal and the default is 5000. If you specify OUTWARN and only one decimal value, the system uses the value as the first limit and assigns the default (5000) for the second value.
OUTLIM(*n1*)	OUTLIM specifies the maximum number of records a user can transmit before the transmission is terminated. ***n*** is a decimal integer whose default is 30000.
VIO(*unitname*)	VIO specifies the device type on which temporary space can be allocated to use by the TRANSMIT and RECEIVE commands. If you do not specify VIO, the system defaults to the UNIT specification for a user in the UADS. If there is no UNIT specification, the system defaults to an installation-defined default or to the system default, SYSALLDA. ***unitname*** is the name of the device type and may be either an esoteric name (SYSDA) or a specific DASD device (3350).
LOGSEL(*logselector*)	LOGSEL the default for the middle qualifiers for the log data set name. ***logselector*** is 1-8 alphameric name of the middle qualifier. The first character must be alphabetic or national. The names must be separated by a period.
LOGNAME(*lognamesuffix*)	LOGNAME specifies the default suffix qualifiers for the log data set name. ***lognamesuffix*** is the name of the qualifiers and must be 1-8 alphmeric characters beginning with an alphabetic or national character. The names must be separated by a period.
USRCTL(*name*)	USRCTL is the name for the NAMES data set. The value for ***name*** must be 1-8 alphameric characters beginning with an alphabetic or national character. The names must be separated by a period.
SYSCTL(*datasetname*)	SYSCTL specifies an alternate NAMES data set. You can use this parameter to specify the NAMES of a data set that contains a set of all of the nicknames at the installation. ***datasetname*** identifies the name of the data set and must be 1-8 alphameric characters beginning with an alphabetic or national character. The names must be separated by periods. The total characters in the names including the periods cannot exceed 44 characters.
SYSOUT(*sysoutclass*)	SYSOUT allows you to specify the default SYSOUT class for messages that are written from utility programs. ***sysoutclass*** identifies the sysout class and may be A-Z, 0-9, and asterisk (*).
DAPREFIX(tuprefix|userid)	DAPREFIX specifies how the control and log data sets will be prefixed when NOPREFIX is specified in the profile. ***tuprefix*** is the prefix that will be used with the TSO user prefix for the control and log data sets. ***userid*** is the prefix that will be used with the userid for the control and log data sets. ***tuprefix*** is the default.

Parameter	Meaning and Use
SEND	Specifies the installation's defaults for the TSO/E SEND command. The defaults are shown here.
OPERSEND(ON/OFF),	OPERSEND specifies whether or not users authorized to use the OPERATOR command can issue the SEND subcommand to send messages or notes. See *TSO/E Commands*
USERSEND(ON/OFF),	USERSEND specifies whether or not users can issue the SEND command to send messages or notes to other terminal users.
SAVE(ON/OFF),	SAVE specifies whether or not the SEND command processor and the OPERATOR SEND command processor are to save messages in a log that the installation specifies.
CHKBROD(ON/OFF)	CHKBROD indicates whether LISTBC processing is to check both the SYS1.BRODCAST and the installation's log for messages.
	ON indicates that the LISTBC is to check both SYS1.BRODCAST and the installation-specified log.
	OFF indicates that the LISTBC processing is to check only the installation specified log for saved messages.
LOGNAME(DSNAME/SYS1.BRODCAST)	LOGNAME identifies the log name of the data set where the system saves messages and notes. ***DSNAME*** identifies one or more lower level qualifiers of the installation specified log. The system automatically sets the first level qualifier to the user's ID. Each qualifier in ***DSNAME*** may be one to eight characters.
	For more information on setting the installation's defaults with the SEND parameter, see *TSO/E Customization*

Member Name: IPCSPRxx (Interactive Problem Control System)

Use of the Member

An IPCSPRxx member contains parameters used during an IPCS session. These session parameters allow an installation to tailor IPCS sessions to its particular requirements. The parameters define the names of various data sets and default values to be used throughout that IPCS session. When you execute the IPCS command, IPCS initialization processes these parameters. Thus, you cannot modify or respecify them during an IPCS session.

The installation may have several IPCSPRxx members. You then specify the member that suits your needs for a particular IPCS session.

Parameter in IEASYSxx (or specified by the operator):

None

The PARM(xx) keyword on the IPCS TSO command specifies the IPCSPRxx member that the system will use during that IPCS session.

Syntax Rules

IPCS processes each statement as lines of TSO Command Language. See *TSO Commands* and *TSO User's Guide*.

IBM-Supplied Defaults

The IPCS installation package includes the default member IPCSPR00.

Internal Parameters

Parameter	Meaning and Use	Value Range	Default
DSD(dsn)	Specifies the data set name of the IPCS data set directory. dsn must be fully qualified and need not be in apostrophes. IPCS appends nothing to either end of the specified name. This parameter is required. Without it, IPCS terminates.		
NODSD	Suppresses the use of problem and data management. This option requires the specification of the NOPDR option.		NODSD
PDR(dsn)	Specifies the data set name of the IPCS problem directory. dsn must be fully qualified and need not be in apostrophes. IPCS appends nothing to either end of the specified name. This parameter is required. Without it, IPCS terminates.		
NOPDR(dsn)	Suppresses the use of problem and data set management. This option requires the specification of the NODSD option.		NOPDR
PROBIDPREFIX(prefix)	Specifies the three-character value used to form a problem identifier where prefix is three alphameric characters. IPCS uses this prefix whenever an IPCS subcommand displays or prints a problem identifier. To form the complete problem identifier, IPCS concatenates a 5-digit decimal number to this prefix. This parameter is required. Without it, IPCS terminates.		
SYSTEM(system-id)	Specifies the default system identifier where system-id is one to eight alphameric characters. The ADDPROB subcommand of IPCS uses this value if you omit the SYSTEM keyword on that subcommand. This parameter is optional. The default is blank.		
GROUP(group-id)	Specifies the default group identifier. group-id is one to eight alphameric characters. The ADDPROB subcommand of IPCS uses this value if you omit the GROUP keyword on the subcommand. This parameter is optional. The default group-id is blank.		
ADMINAUTHORITY(userid-list)	Specifies the TSO userids of the persons with IPCS administrative authority. userid-id is one to eight alphanumeric characters. The owners of the specified TSO userids have the same authority that a problem owner has, but they have that authority over **all** problems in the problem directory. Specifying administrators does not affect the privileges of a problem owner. An owner can still modify and delete problems that he owns. Specifying persons with administrative authority provides centralized control over all problems defined to IPCS. Administrators can assign and reassign problem owners, access a problem when the owner is unavailable, update problems when their attributes change, correct attributes that were incorrectly specified, etc. This parameter is optional. If not specified, no one has administrative authority for the installation. If specified, it is a list whose entires are separated by one or more blanks, commas, or horizontal tabulation characters (X'05'). Each entry in the list may name the userid of an administrator or may designate an inclusive range of userids. The following example authorizes the person with userid THEBOSS plus all persons with userids beginning with the letters ADMIN as administrators. *ADMINAUTHORITY(THEBOSS,ADMIN:ADMIN)*		

Parameter	Meaning and Use	Value Range	Default
DELETEAUTHORITY(userid-list)	Specifies the TSO userids of the persons with IPCS delete authority. The owners of the specified TSO userids are the **only** persons who can delete problems. Specifying persons with delete authority diminishes the privileges of a problem owner and the person with administrative authority. They can modify problems but can not delete them. Specifying persons with delete authority provides centralized control over removing problems from IPCS. When a problem is deleted, all the accumulated information about it is lost. By designating persons who can delete problems, you can help prevent the inadvertent loss of such information. This parameter is optional. If specified, only the designated person can use the DELPROB subcommand of IPCS. Problem owners cannot use the DELPROB subcommand nor can the person designated with the ADMINAUTHORITY parameter (if the userid specified is the different.) If not specified, problem owners and the person specified with the ADMINAUTHORITY parameter can use the DELPROB subcommand. If specified, it is a list whose entries are separated by one or more blanks, commas, or horizontal tabulation characters (X'05'). Each entry in the list may name the userid of an administrator or may designate an inclusive range of userids.		
LINELENGTH(value)	Specifies the default record length (LRECL) for the IPCS print output data set. *value* is a two or three decimal digit ranging from 83 through 255. This parameter is optional. If you do not specify an LRECL for the print output data set and if it is not specified in the session parameters, the default is 137.	83-255	137
PAGESIZE(value)	Specifies the default number of lines per page for the IPCS print output data set. *value* is a decimal number ranging from 3 through 2^{31}-1. This parameter is optional. If you do not specify it, the default is 60.	See text	60

Member Name: LNKLSTxx (Link Library List)

Use of the Member

LNKLSTxx contains the names of program libraries that are to be concatenated to SYS1.LINKLIB to form the LNKLST concatenation. The default member LNKLST00, built at sysgen, contains only the name SYS1.LINKLIB. The installation may add other library names to LNKLST00 through use of the IEBUPDTE utility.

You can create any number of LNKLSTxx members. NIP opens and concatenates each library in the order in which library names are listed, starting with the first-specified LNKLSTxx member.

The number of data sets in the LNKLST concatenation is limited. The actual limit of data sets in the LINKLIB concatenation is controlled by the number of extents allowed in the DEB. Use the following algorithm to determine whether a given LNKLST concatenation can be built:

2041 (32) + (16n) + (k-1)

where:

n = the number of DASD extents

k = the number of data sets in the LNKLST (including SYS1.LINKLIB)

There can be up to 118 single-extent data sets, or 125 extents in 8 data sets in the LNKLST.

Library Lookaside (LLA): The LLA address space enhances system performance by eliminating the I/O required by BLDL on the LNKLST concatenation. LLA replaces the fixed or pageable BLDL table with a refreshable hashed directory table for the entire LNKLST concatenation. LLA thus eliminates both the performance degradation related to LINKLIB directory searches and the necessity of tuning the BLDL table and LINKLIB concatenation.

LLA provides for dynamic additions, deletions, or updates to members of libraries managed by LLA. LLA dynamically accumulates statistics for each LNKLST module that is fetched. Using these and other statistics, LLA asynchronously places (stages) the modules in data areas that the virtual lookaside facility (VLF) manages. You, can implement modify LNKLST libraries dynamically by refreshing LLAs directory table. To refresh LLA's directory table, either issue the MODIFY LLA REFRESH command, or stop and subsequently restart LLA. LLA also allows you to update specific entries in the table by issuing the MODIFY LLA UPDATE command.

During IPL, NIP opens the LNKLST concatenation. NIP creates a data extent block (DEB) that describes the data sets concatenated to LINKLIB and the extents of SYS1.LINKLIB. These SYS1.LINKLIB extents described in the DEB remain in effect for the duration of the IPL.

After NIP, Library Lookaside (LLA) is started. LLA manages the LNKLST performance and can be used to control updates to the LNKLST libraries. Refer to *Starting Library Lookaside (LLA)* and *Modifying the Library Lookaside (LLA) Directory* previously in this chapter for more information about LLA.

You should allocate LNKLST data sets exclusively with primary extents; otherwise, updates to LNKLST libraries could cause the data set to expand into secondary extents. All subsequent requests that access a module in a secondary extent abend with an I/O error. The abend occurs because the directory entry references extents outside those described in the DEB.

If LNKLST data sets have expanded into secondary extents since the last IPL, you can access modules residing in those extents by:

- Using a steplib, joblib, or tasklib.
- Compressing the data set and then refreshing LLA. If the compress restores the data set to its primary extent allocation, no errors should occur when accessing the individual members of the data set.

If neither of the preceding options is possible, you must re-IPL to reopen the LNKLST data sets and to reconstruct the DEB.

Once the system has been IPLed, do not add or delete data sets from the LNKLST concatenation. LLA cannot access the directories of data sets that have been added to the LNKLST concatenation once the system has IPLed. If data sets are deleted from the LINKLIB concatenation after IPL, an attempted refresh of LLA abends with a protection exception.

Notes:

1. SYS1.LINKLIB is automatically APF-authorized. Additionally, if the default for the LNKAUTH system parameter is taken (LNKAUTH=LNKLST) or is specified in IEASYSxx or by the operator, any library concatenated to SYS1.LINKLIB is automatically APF-authorized when accessed as part of the concatenation.
2. Libraries in the LNKLST concatenation must be cataloged in the system master catalog. (OS CVOLs and user catalogs are not searched for LNKLST libraries.)

Parameter in IEASYSxx (or specified by the operator):

LNK= { **aa** | **(aa,bb,...[,L])** | **(,L)** }

The two alphameric characters, represented by aa (or bb, etc.), are appended to LNKLST to identify one or more LNKLSTxx members of parmlib. If the parameter is not specified either in IEASYSxx or by the operator, the default member LNKLST00 is used.

If the 'L' keyword is specified - either in parmlib or from the operator's console - NIP generates a list of the names of the program libraries that are concatenated to SYS1.LINKLIB. This list is then printed at the operator's console preceded by message "IEA331I LINK LIBRARY CONCATENATION."

Syntax Rules

The following rules apply to the creation of LNKLSTxx by means of the IEBUPDTE utility:

- Place on each record a string of data set names separated by commas.
- Indicate continuation by placing a comma followed by at least one blank after the last name on a record.
- If you place the name SYS1.LINKLIB on any record in any LNKLSTxx member, the name will be ignored.
- Be careful not to specify the same library name more than once in a succession of LNKLSTxx members. The same library will be concatenated as many times as it appears in all specified LNKLSTxx members. The result can be degraded performance because the system can search the same library more than once for a given module.

Syntax Example:

```
IEASYSxx:   ...,LNK=(00,01,02,03)
LNKLST00:   SYS1.CMDLIB,SYS1.TSORTNS,SYS1.BTAMLIB
LNKLST01:   SYS1.LINKLIB,SYS1.U30LIB,SYS2.U30LIB
LNKLST02:   SYS1.AUXLIB,SYS1.JES3
LNKLST03:   SYS1.TEST
```

The result of the foregoing specification is that the following data sets, in the order specified, are concatenated to SYS1.LINKLIB:

SYS1.CMDLIB,SYS1.TSORTNS,SYS1.BTAMLIB,SYS1.U30LIB,
SYS2.U30LIB,SYS1.AUXLIB,SYS1.JES3,SYS1.TEST

Note that, in the example, the specification of SYS1.LINKLIB (in LNKLST01) was ignored.

IBM-Supplied Defaults

Default member LNKLST00 contains only the name SYS1.LINKLIB.

Internal Parameters

Not applicable

Member Name: LPALSTxx (LPA Library List)

Use of the Member

LPALSTxx contains the names of program libraries that you want the system to concatenate to SYS1.LPALIB. This concatenation is referred to as the LPALST concatenation, which the system uses to build the pageable LPA (PLPA).

You can create any number of LPALSTxx members. NIP opens and concatenates each library in the order in which library names are listed, starting with the first-specified LPALSTxx member.

The number of data sets in the LPALST concatenation is subject to the same limitation as any other data set concatenation. If there are more data sets specified than can be described in the data extent block (DEB) control block, the system truncates the concatenation and issues messages that indicate which data sets were not included in the concatenation.

If an LPALSTxx member exists, and an LPALST concatenation is successfully opened, the system uses the LPALST concatenation to build the PLPA. Otherwise, the system builds the PLPA from the modules named in SYS1.LPALIB. There is no default LPALSTxx member. However, SYS1.LPALIB acts as the default in that the system uses SYS1.LPALIB to build the PLPA if the installation has not created an LPALSTxx member.

Notes:

1. The system considers SYS1.LPALIB to be APF authorized when it is opened as part of the LPALST concatenation. All data sets that you want to concatenate to SYS1.LPALIB must also be APF authorized. Use the IEAAPFxx parmlib member (and the APF system parameter) to authorize the data sets that you concatenate to SYS1.LPALIB.
2. Libraries in the LPALST concatenation must be cataloged in the system master catalog. (OS CVOLs and user catalogs are not searched for LPALST libraries.)

Parameter in IEASYSxx (or specified by the operator):

LPA = { aa | (aa,bb,...[,L]) }

The two alphameric characters, represented by aa (or bb, etc.), are appended to LPALST to form the name of the LPALSTxx parmlib member(s). If the L option is specified, the system displays (at the operator's console) the names of the data sets successfully concatenated to SYS1.LPALIB.

Syntax Rules

The following rules apply to the creation of LPALSTxx by means of the IEBUPDTE utility:

- On each record, place a string of data set names separated by commas.
- Indicate continuation by placing a comma followed by at least one blank after the last data set name on a record.
- If you place the name SYS1.LPALIB on any record in any LPALSTxx member, the name will be ignored.
- Be careful not to specify the same data set name more than once in a succession of LPALSTxx members. The same data set name will be concatenated as many times as it appears in all specified LPALSTxx members. This could cause degraded system performance because the system would search the same data set more than once for a given module.

Syntax Example:

```
IEASYSxx:   ...,LPA=(00,01,02,03)
LPALST00:   SYS1.CMDLIB,SYS1.TSORTNS,SYS1.BTAMLIB
LPALST01:   SYS1.U30LIB,SYS2.U30LIB,SYS1.LPALIB
LPALST02:   SYS1.AUXLIB,SYS1.JES3
LPALST03:   SYS1.TEST
```

The result of the foregoing specification is that, in the order specified, the following data sets are concatenated to SYS1.LPALIB:

SYS1.CMDLIB,SYS1.TSORTNS,SYS1.BTAMLIB,SYS1.U30LIB,SYS2.U30LIB,
SYS1.AUXLIB,SYS1.JES3,SYS1.TEST

Note that, in the example, the specification of SYS1.LPALIB (in LPALST01) was ignored.

IBM-Supplied Default

There is no default LPALSTxx member. If the installation does not create an LPALSTxx member (and therefore no data sets are concatenated to SYS1.LPALIB), the system uses only SYS1.LPALIB to build the PLPA.

Internal Parameters

Not applicable

Member Name: MPFLSTxx (Message Processing Facility List)

Use of Member

MPFLSTxx contains information that the message processing facility (MPF) uses to control message display and message processing.

Message display refers to color, highlighting, and intensity attributes that the system uses when displaying messages on an operator console. You can specify these attributes in the MPFLSTxx member; of course, the console must support the attributes you specify. **Message processing** refers to message suppression, message retention, message automation eligibility, and using installation-supplied WTO/WTOR exits to process messages.

This description of the MPFLSTxx parmlib member has the following order:

- General information including syntax rules and selecting a MFPLSTxx member.
- Controlling message display using MPFLSTxx.
- Controlling message processing using MPFLSTxx.

Parameter in IEASYSxx:

None

General Syntax Rules

These rules apply to the creation of MPFLSTxx by means of the IEBUPDTE utility:

- Each record is 80 characters long. The system ignores columns 72 through 80 and leading blanks.
- Each message processing record can contain only one message identifier.
- Begin comments with /* in any column. It is best to end the comment with a */. If, however, you do not include the end delimiter, the system recognizes the next statement.

Selecting an MPFLSTxx member

You can select a particular MPFLSTxx parmlib member in one of the following ways:

- Specifying the INIT statement with the MPF keyword in the CONSOLxx parmlib member. CONSOLxx is used at system initialization.
- Including the SET MPF command in the COMMNDxx parmlib used at system initialization.
- Issuing the SET MPF command after system initialization.

Two alphanumeric characters are appended to MPFLST to form the name of the MPFLSTxx parmlib member.

Note: Select the MPFLSTxx parmlib member in **either** the CONSOLxx member **or** in the COMMNDxx member. Do not maintain this selection in both members. It is best to maintain the MPFLSTxx member selection in the CONSOLxx member and delete it from the COMMNDxx member. After initialization, you can issue the SET MPF command to change the MPFLSTxx member; however, this change is temporary. At the next IPL, the system uses the MPFLSTxx member specified in the CONSOLxx or COMMNDxx member. See the information on the CONSOLxx and COMMNDxx parmlib members.

If you issue the SET MPF=NO command, the system uses the IBM-supplied defaults for message display and for message processing. See *System Commands* for descriptions of the SET command.

IBM-Supplied Parmlib Member

None.

Controlling Message Display through MPFLSTxx

Message display refers to color, highlighting, and intensity attributes that the system uses when displaying messages on an operator console. You can specify these attributes on the **.MSGCOLR** statement in the MPFLSTxx member; of course, the console must support the attributes you specify. On the .MSGCOLR statement, you can indicate the display attributes that the system is to use in one of the following ways:

- IBM-supplied defaults (DEFAULT)
- Attributes specified in the previous MPFLSTxx member (NOCHANGE),
- Color (c), highlighting (h), and intensity (i) attributes specified within this MPFLSTxx member for a message area (msgarea).

Syntax for Controlling Message Display

The syntax for the **.MSGCOLR** statement is:

```
                 ┌ DEFAULT                                    ┐
.MSGCOLR         │ NOCHANGE                                   │   [/*comments*/]
                 └ msgarea(c,h[,i])[,msgarea(c,h[,i])]...     ┘
```

IBM-Supplied Defaults for .MSGCOLR

If you do not specify a .MSGCOLR statement, the system uses IBM-supplied defaults. See the following description on the .MSGCOLR statement for these defaults.

If you specify a .MSGCOLR statement with no operands, the system defaults to NOCHANGE. NOCHANGE indicates that the color, highlighting, and intensity attributes are to be the same as those specified in the previous MPFLSTxx member.

Listing the Message Display Attributes for the Current MPFLSTxx

You can use the DISPLAY MPF command to list the current color intensity and highlighting specifications for a specific console or for all consoles. See *System Commands* for descriptions of the DISPLAY MPF command.

MPFLSTxx Parameter for Controlling Message Display

Statement	Operand	Meaning and Use
.MSGCOLR		.MSGCOLR indicates the beginning of a statement that defines the display attributes for messages.
	DEFAULT	DEFAULT causes the system to use the IBM-supplied color, highlighting, and intensity defaults. See the following description on *msgarea*. (Figure 3-8 on page 3-197 lists the defaults.) If you need to specify a .MSGCOLR DEFAULT statement, specify only one. If the first .MSGCOLR statement specifies DEFAULT, the system ignores any subsequent .MSGCOLR statements, issues an error message, and continues processing any remaining statements.
	NOCHANGE	NOCHANGE indicates that the color, highlighting, and intensity attributes are to be the same as those specified in the previous MPFLSTxx member. If you need to specify a .MSGCOLR NOCHANGE statement, specify only one. If the MPFLSTxx member does not contain a .MSGCOLR statement, the system defaults to NOCHANGE. If the first .MSGCOLR statement specifies NOCHANGE, the system ignores any subsequent .MSGCOLR statements, issues an error message, and continues processing any remaining statements. The system uses the color, highlighting, and intensity attributes specified in the previous MPFLSTxx member.
	msgarea(c,h[,i])	*msgarea* specifies the message area (message type or field); contain the accepted values and the defaults for *msgarea*. This operand allows you to specify the color (*c*), highlighting (*h*), and intensity (*i*) attributes for a message area. Color attributes (*c*) are specified as follows: R-Red W-White G-Green B-Blue P-Pink Y-Yellow T-Turquoise Highlighting attributes (*h*) are specified as follows: N-Normal (colored characters on a black background) B-Blinking R-Reverse video (black characters on a colored background) U-Underscored characters Intensity attributes (*i*) are specified as follows: N-Normal intensity H-High intensity If you specify a *msgarea*, you must specify the color and highlighting attributes. If you do not, or if the system detects an error in any specified value, it issues an error message, stops checking the statement in error, and continues processing any subsequent statements. The intensity attribute, however, is optional. If you do not specify the intensity attribute, the system uses the IBM-supplied defaults. If you specify multiple .MSGCOLR statements for the same msgarea, the system uses the last valid statements.

Figure 3-8. Values for msgarea

Values for msgarea	Meaning and Use	Defaults (c,h,i)
ENTRYARA	Specifies attributes for the entry area.	(G,N,N)
EVETACTN	Specifies attributes for eventual action messages (issued with a descriptor code of 3).	(G,N,N)
GENMSG	Specifies attributes for general system messages (issued with a descriptor code other than 1, 2, 3 or 11). **Note:** The system displays message IEE152I on the instruction line. The message identifier, however, is displayed using the GENMSG attributes. The message text is displayed using the SELPEN attributes.	(G,N,N)
IMEDACTN	Specifies attributes for immediate action messages (issued with a descriptor code of 2 or WTOR messages). **Note:** For a device with limited image capability (such as the 3290), the reverse video highlighting attribute applies only to the action characters * or @ that precedes the message identifier.	(W,N,H)
INSTRERR	Specifies attributes for error messages that appear in the instruction line.	(W,N,H)
OOLCNTL	Specifies attributes for the control line text of an out-of-line status display, not including the selector pen detectable fields on the control line. The console id and area id, cca, in the control line are displayed using these attributes.	(T,N,N)
OOLDATA	Specifies attributes for the data lines of an out-of-line status display.	(G,N,N)
OOLLABEL	Specifies attributes for the label lines of an out-of-line status display.	(T,N,N)
PPMSG	Specifies attributes for non-action messages issued by a problem program (issued with a descriptor code other than 1, 2, 3, or 11).	(G,N,N)
SELPEN	Specifies attributes for fields that are selector pen detectable, such as: in the instruction line - ENTER CANCEL D C,K in the control line of an out-of-line display - F E in the control line of a TRACK display - U PT H in the control line of an in-line display - C	(B,N,N)
URGATTN	Specifies attributes for urgent attention messages (issued with a descriptor code of 1 for system failure or 11 for critical eventual action). **Note:** For a device with limited image capability (such as the 3290), the reverse video highlighting attribute applies only to the action characters * or @ that precedes the message identifier.	(R,N,H)
WARNLGEN	Specifies attributes for the left half of the warning line for general messages such as: IEE163I MODE=R IEE163I MODE=RD IEE161I WARNING-CON=N, DEL=Y	(B,N,N)
WARNRGEN	Specifies attributes for the right half of the warning line for general messages such as: IEE160I UNVIEWABLE MESSAGE	(B,N,N)
WARNRURG	Specifies attributes for the right half of the warning line for urgent attention messages such as: IEE159E MESSAGE WAITING	(R,B,H)

Note: If you specify EVETACTN, GENMSG, OOLDATA, OOLLABEL, or PPMSG for a device with limited image capability (such as the 3290), the system ignores the reverse video highlighting attribute.

Controlling Message Processing through MPFLSTxx

MPFLSTxx allows you to specify how you want to respond to process WTO/WTOR messages. **Message processing** refers to message suppression, message retention, message automation eligibility, and the use of installation-supplied WTO/WTOR exits.

Message suppression means that the message is logged in the hardcopy log, but it does not appear at an MVS operator's console. Command responses can not be suppressed.

Message retention means that action or WTOR messages are saved by the action message retention facility (AMRF) on the action message retention queue. Retention allows the operator to view the message later. If you choose not to retain a message, the system will not add it to the action message retention queue.

Message automation eligibility means that you are using an automation subsystem to process particular WTO/WTOR messages in a pre-determined way. The automation subsystem, such as NetView™ allows the installation to program the processing for a particular message. For example, your installation may be using an automation subsystem to

- modify that text of the message,
- re-route a message to a different operator's console or to a different system for processing,
- record some information contained in the message text for future use,
- respond to the message.

An automation subsystem can look at the message text and the message attributes and perform the programmed action. You can use MPFLSTxx to identify whether or not you want a message to be eligible for automation processing. The automation subsystem must then select and process the messages. You can also use MPFLSTxx to indicate information to pass to an automation subsystem. Messages that are eligible for automation processing can also be suppressed or retained. Message suppression and retention are functions separate from automation.

Note: Operator Communication Control Facility (OCCF) does not recognize the message automation eligibility specification. OCCF processes only unsuppressed messages. Beginning with NetView Release 2, the NetView subsystem recognizes the message automation eligibility specifications in MPFLSTxx. For more details, see *Automated Operations Planning Guide* and *NetView Administration Reference*.

Your installation may have installation-written exits to handle WTO/WTOR messages. Through MPFLSTxx, you can specify which messages are to go to which exit. This exit can examine the message text and the message attributes and decide whether to suppress, retain, or take other actions on the message. This exit can also examine and modify the token.

Specifying Message Processing

For the message(s) that you specify in MPFLSTxx, you can indicate that the system is to:

- Suppress one or more specific messages or all messages that begin with a particular prefix. SUP(YES/NO) provides this option.
- Retain action messages (descriptor codes 1, 2, 3, 11) using the action message retention facility (AMRF). RETAIN(YES/NO) provides this function.
- Allow an automation subsystem, such as NetView, to respond to the WTO/WTOR message(s) that you specify. AUTO(YES/NO/token) allows you to identify whether or not a message is eligible for automation processing and to pass information (a token) to the automation subsystem.
- Pass control to an installation-supplied WTO/WTOR exit to process the message(s). The USEREXIT option allows you to identify the name of the exit.
- Assign user-specified defaults for specific messages that are identified in the MPFLSTxx member. The .DEFAULT statement allows you to change the system-assigned message processing defaults for any message you specify.
- Assign user-specified defaults for messages that are not identified in the MPFLSTxx member. The .NO_ENTRY statement allows you to indicate how the system is to handle messages that are not specified in the MPFLSTxx member.
- Indicate or flag the beginning of particular messages that are written to the JES3 hard copy log. The MPFHCF statement allows you to specify the character you want to use as an indicator.

Exceptions to Specifying Message Processing Attributes

You cannot use an MPFLSTxx parmlib member to directly suppress any of the following:

- Command responses where MCSFLAG=RESP was specified on the WTO.
- Command responses with descriptor code 5 (except those issued in response to the MONITOR command).

Syntax for Controlling Message Processing

To control message processing, MPFLSTxx recognizes one parameter, **msgid**, and four statements: (**.DEFAULT**, **.NO_ENTRY**, **MPFHCF**, and **.MSGIDS**)

- **msgid** allows you to specify a particular message id or prefix. This specification is also called the message processing record.
- **.DEFAULT** allows you to specify the defaults for groups of messages listed in the MPFLSTxx parmlib member.
- **.NO_ENTRY** allows you to specify the default processing you want for messages that are **NOT** identified in the MPFLSTxx parmlib member. (Note that you must code the underscore (_) in the statement.)
- **MPFHCF** allows you to indicate or flag messages that JES3 writes to the hardcopy log. Note that this statement does not begin with a period (.).
- **.MSGIDS NOCHANGE** allows you to specify that the messages identifiers and the message processing are to be the same as those specified in the previous MPFLSTxx member.

The syntax of the **message processing record (msgid)** is:

```
msgid  [ ,AUTO  [ (YES)   ] ,RETAIN  [ (YES) ] ,SUP [ (YES) ] ]
       [        [ (NO)    ]          [ (NO)  ]     [ (NO)  ]  ]
       [        [ (token) ]                                    ]
       [                                                       ]
       [ [,USEREXIT(exitname)]       [ /*comments*/]           ]
```

The syntax of the **.DEFAULT** statement is:

```
.DEFAULT [ ,AUTO  [ (YES)   ] ,RETAIN  [ (YES) ] ,SUP [ (YES) ] ]
         [        [ (NO)    ]          [ (NO)  ]     [ (NO)  ]  ]
         [        [ (token) ]                                    ]
         [                                                       ]
         [ [,USEREXIT(exitname)]       [ /*comments*/]           ]
```

The syntax of the **MPFHCF** statement is:

```
MPFHCF=[x/&]
```

The syntax for the **.MSGIDS** statement is:

```
.MSGIDS NOCHANGE
```

The syntax of the **.NO_ENTRY** statement is:

```
.NO_ENTRY [ ,AUTO  [ (YES)   ]   ,RETAIN  [ (YES) ] ]
          [        [ (NO)    ]            [ (NO)  ]  ]
          [        [ (token) ]                       ]
          [                                          ]
          [ ,SUP  [ (YES) ]   [/*comments*]          ]
          [       [ (NO)  ]                          ]
```

IBM-Supplied Defaults for Message Processing

If you do not specify any message processing statements in the MPFLSTxx member, the defaults are AUTO(YES), RETAIN(YES), and SUP(NO). The system considers all messages eligible for automation processing, and it retains action and WTOR messages, and displays all messages.

Listing the Message Processing Attributes for the Current MPFLSTxx

You can use the DISPLAY MPF command to list the results of the current MPFLSTxx member. The DISPLAY MPF command displays:

- The messages being suppressed by MPF
- The action messages *not* being retained by the action message retention facility
- The user exits that receive control for selected messages
- The status of the general WTO user exit, IEAVMXIT.

See *System Commands* for descriptions of the DISPLAY command.

Note: The DISPLAY MPF command will not display whether a message is eligible for automation processing. The hardcopy log, however, will contain the automation tracking indicator.

To diagnose MPF processing, you can also use the hardcopy log to determine the message processing options used for a message.

Using Other Methods to Suppress Messages

WTO/WTOR messages can be suppressed through an MPFLSTxx member and by other methods. When an installation uses more than one method, message suppression is performed in the following order:

1. The WTO/WTOR user exit, IEECVXIT, can be used by the installation to suppress some messages. See *User Exits* for information.
2. The message processing facility (MPF) itself suppresses messages.
3. An installation-supplied WTO/WTOR user exit (either IEAVMXIT or an exit you name on the USEREXIT parameter in MPFLSTxx) can suppress messages. (Note that the exit can also override MPF suppression. That is, the exit can prevent MPF from suppressing a message.)
4. An active subsystem (such as JES2, JES3, or NetView) can suppress messages.
5. CONTROL V command can suppress messages by specifying only the message levels that are to be displayed at a console. The LEVEL keyword on the CONSOLE statement in the CONSOLxx member of SYS1.PARMLIB controls the message levels for the console.

Internal Parameters

Parameter	Options	Meaning and Use
msgid		Identifies a message or group of messages to be processed. ***msgid*** consists of either: • A complete message identifier of one to ten characters • A message prefix, which is a portion of the message identifier, followed by an asterisk(*) A message identifier begins with the first non-blank character of the message text and continues until the next blank. You can specify only one msgid per record. If a msgid is repeated within an MPFLSTxx member, the system uses the options specified on the first record and ignore the duplicates. If you want MPF to process a specific message, you must specify the complete message id, for example IEF124I. If you want MPF to process all messages that begin with a specific prefix, you can specify the prefix and an asterisk, such as IEF24*. Use the message prefix with an asterisk to define message suppression carefully. Too wide a suppression, such as IEF*, could suppress messages that you need for effective system operation.
	,AUTO(YES/NO/token)	Specifies whether or not the message (***msgid***) is eligible for processing by an automation subsystem, such as NetView. AUTO(YES) makes the message eligible for automation processing. AUTO(NO), which is the system default, makes the message ineligible for automation processing. If you specify ***msgid*** and do not specify the AUTO option, the system defaults to NO. If, however, you do not specify any message processing statements for a message, the system considers the message eligible for automation processing AUTO(YES). To change the system default, use the .DEFAULT or the .NO_ENTRY statement. The *token* value is available for MPF user exit processing and for processing by an automation subsystem. Specifying a token indicates that the message is eligible for for processing by the automation subsystem. The token value must be 1 to 8 alphanumeric characters. You may not use a left parenthesis "(" as part of the token value. Imbedded blanks are allowed in the token value. For example, if you code an 'N,O, and a blank', AUTO(NO), the system takes the "NO" as a token value. A token value is not recognized before MVS/SP 2.2.0.
	,RETAIN(YES/NO)	If the message identified by the ***msgid*** is an action message or WTOR, RETAIN specifies whether or not the message is to be retained by the active message retention facility (AMRF). AMRF can retain only action messages and WTORs. The system default is RETAIN(YES). You can use the .DEFAULT statement followed by a msgids to change the system default for the list of messages. To view a retained message, use the DISPLAY R command.
	,SUP(YES/NO)	Specifies whether or not MPF is to suppress the message identified by ***msgid***. The system default is SUP(YES). If you specify a ***msgid*** without the SUP option, the system does not display the message, SUP(YES). If, however, you do not specify any message processing statements for a message, the system displays the message, SUP(NO). To change the system defaults, use the .DEFAULT or the .NO_ENTRY statement.

Parameter	Options	Meaning and Use
	,USEREXIT(*exitname*)	Specifies the name of an installation-supplied WTO/WTOR user exit routine that is to get control each time the system issues the message(s). This routine can process the message(s); it can suppress, retain, or respond to a message. It can make the message eligible for automation processing, as well as, take other actions on the message. The ***exitname***, can be from one to eight alphanumeric (A-Z, 0-9) and national characters (&, *, $). The first character must be alphabetic or numeric. If you do not specify an exitname, the system uses IEAVMXIT, if it exists. To change the system default, use the .DEFAULT statement. For more information on the WTO/WTOR user exits, see the CONSOLxx parmlib member in this book, and *User Exits*.
.DEFAULT		.DEFAULT allows you to specify the defaults that you want for the message processing records (msgids) that follow .DEFAULT. The options you specify on the .DEFAULT statement override the system defaults for messages that you list. On the .DEFAULT statement, you can specify that the message is eligible for automation processing, retention, and/or suppression, and the user exit that is to process the message. Through AUTO(token), you can also specify information to be passed to the automation subsystem. .DEFAULT with no options, results in the message being ineligible for automation processing, and indicates that the system is to suppress and retain any listed action or WTOR messages. IEAVMXIT, if it exists, receives that message. You can use the .DEFAULT statement multiple time within an MPFLSTxx member. Each group of messages following a .DEFAULT statement should have common option values. This allows you to control attributes assigned by default to each message id without having to change every message processing record. On a particular message record (msgid statement) that follows a .DEFAULT statement, you can specify specific operand values that override, for that message, the . DEFAULT values. If there are multiple occurrences of a message id listed under one .DEFAULT statement, the system uses the options for the first occurrence and ignores the others. If an MPFLSTxx member contains multiple .DEFAULT statements, the system uses the values on the .DEFAULT statement that precedes the first message record (msgid statement) in that group.
	,AUTO(YES/NO/token)	Specifies whether or not a message or a list of messages following the .DEFAULT statement is eligible for processing by an automation subsystem, such as NetView. AUTO(YES) makes the message(s) eligible for automation processing. AUTO(NO) (the system default for the .DEFAULT statement) makes the subsequent message(s) ineligible for automation processing. The *token* value is available for MPF user exit processing and for processing by an automation subsystem. Specifying a token indicates that the message is eligible for for processing by the automation subsystem. The token value must be 1 to 8 alphanumeric characters. You may not use a left parenthesis "(" as part of the token value. Imbedded blanks are allowed in the token value. For example, if you code an 'N,O, and a blank', AUTO(NO), the system uses the "NO" as a token value. The system does not recognize a token value before MVS/SP 2.2.0. **Note:** If you specify a .DEFAULT and omit the AUTO option, the system will not consider the message(s) eligible for automation processing. If, however, you do not specify any message processing statements for a message, the system considers the message eligible for automation processing.

Statement	Operand	Meaning and Use
	,RETAIN(YES/NO)	RETAIN specifies whether or not the subsequent action message(s) or WTOR is to be retained by the action message retention facility (AMRF). The default is RETAIN(YES). To view a retained message, use the DISPLAY R command.
	,SUP(YES/NO)	SUP specifies whether MPF is to display (NO) or to suppress (YES) the subsequent message(s). The default is SUP(YES). If you specify a .DEFAULT statement without the SUP option, the system suppresses the messages. If, however, you specify the SUP option on a subsequent message record, the system uses the value on the message record. If, however, you do *not* specify any message processing statements for a message, the system default is to display the message.
.DEFAULT (continued)	,USEREXIT(***exitname***)	Specifies the name of an installation-supplied WTO/WTOR user exit routine that is to get control each time the system issues one of the following messages. This routine then processes the message(s). The ***exitname***, can be from one to eight alphanumeric (A-Z, 0-9) or national characters (@, $, *). The first character must be alphabetic or numeric. If you do not specify an *exitname* the system defaults to IEAVMXIT, if it exists. For more information on the WTO/WTOR user exits, see the CONSOLxx member of SYS1.PARMLIB in this book, and *User Exits*.
MPFHCF=	[x/&]	If JES3 writes the messages to the hardcopy log, this parameter specifies the indicator (***x***) used to identify suppressed messages in the hardcopy log. (*JES2 ignores this statement.*) The indicator can be any character, including a blank. If ***x*** specifies an indicator that is not a printable character, the indicator is translated to a blank. When MPFHCF is not specified, the default character is an ampersand (&). **Note:** This statement does NOT begin with a period (.).
.MSGIDS	NOCHANGE	.MSGIDS NOCHANGE specifies that the message identifiers are to be the same as those specified in the previous MPFLSTxx member (the one the system was previously using). Use this statement in conjunction with the .MSGCOLR statement. The .MSGIDS NOCHANGE statement should not be specified in an MPFLSTxx member that includes message identifiers. If syntactically correct message identifiers (msgid statements) precede the .MSGIDS NOCHANGE statement, the system ignores the statement, issues an error message, and continues processing the remaining statements. If a valid .MSGIDS NOCHANGE statement precedes message identifiers, the system ignores the message identifiers, issues an error message, and continues processing the remaining statements. If multiple .MSGIDS NOCHANGE statements occur in one MPFLSTxx member, the system processes each statement as though it were the first statement.

Statement	Operand	Meaning and Use
.NO_ENTRY		.NO_ENTRY specifies the message processing options for all messages that are NOT specified in the MPFLSTxx member. The options you specify on the .NO_ENTRY statement overrides the system defaults for messages that are not specified in this member. On the .NO_ENTRY statement, you can specify whether all messages not specified in the MPFLSTxx member are to be considered eligible for automation processing, retained, and suppressed. .NO_ENTRY is a very powerful statement and should be used with care. **Note:** .NO_ENTRY requires the underscore (_) in the syntax. You may not, however, be able to print the underscore character on your printer. If you specify a .NO_ENTRY statement with no options, the system considers the messages that are not specified in the MPFLSTxx member to be eligible for automation processing, it retains action and WTOR messages, and it displays all messages (AUTO(YES) RETAIN(YES) SUP(NO)). These options are the same defaults that the system uses when no message processing record is specified in an MPFLSTxx member for a particular message. .NO_ENTRY checks the syntax and then ignores the USEREXIT(exitname) statement. IEAVMXIT receives control, if it exists. Specify only one .NO_ENTRY statement in an MPFLSTxx member. If there is more than one occurrence of a .NO_ENTRY statement in an MPFLSTxx member, the system checks the syntax of the duplicate and uses the options on the first .NO_ENTRY statement.
	,AUTO(YES/NO/token)	For all messages NOT identified in the MPFLSTxx, AUTO indicates whether or not the message is eligible for processing by an automation subsystem, such as NetView. AUTO(YES), the system default makes the messages eligible for automation processing. AUTO(NO) makes the messages ineligible for automation processing. The *token* value is available for MPF user exit processing and for processing by an automation subsystem. Specifying a token indicates that the message is eligible for for processing by the automation subsystem. The token value must be 1 to 8 alphanumeric characters. You may not use a left parenthesis "(" as part of the token value. Imbedded blanks are allowed in the token value. For example, if you code an 'N,O, and a blank', AUTO(NO), the system uses the "NO" as a token value. A token value is not recognized before MVS/SP 2.2.0. **Note:** IEAVMXIT, if it exists, gets control for all messages whose message ids are not specified in the MPFLSTxx member.
	,RETAIN(YES/NO)	For all action or WTOR messages not identified in MPFLSTxx, RETAIN specifies whether or not the message is to be retained by the action message retention facility (AMRF). The default is RETAIN(YES). To view a retained message, use the DISPLAY R command.
	,SUP(YES/NO)	SUP specifies whether or not MPF is to suppress the message. The default is to display the messages. ***WARNING:*** SUP(YES) causes the system to suppress ALL messages whose ids are NOT identified in the active MPFLSTxx member. This may result in most or all messages being suppressed from the MCS console. This action is useful in certain situations, such as a remote system, but it can be detrimental if an operator is expecting these messages.

Approaches to Message Suppression Using MPFLSTxx

You can use MPFLSTxx for message suppression conservatively or aggressively. In a *conservative* environment, you would suppress messages by identifying the complete ids. You would suppress, however, only messages regarded from an operator's perspective as useless.

In an *aggressive* environment, you would suppress messages through the conservative approach and you would possibly choose to suppress all messages from certain components. If you decide to suppress all of the messages from a specific component, make sure that none of the messages have any vital importance to the operator or to OCCF. (Suppressing a message does not effect processing by the NetView subsystem.) In an aggressive approach, you could also choose to suppress messages that rarely are important to the operator.

The following pages contain five lists of suppressible messages. ***They are examples that you should review according to the environment in which you might use them. Make sure that you review each message and determine whether your installation should suppress it or not.*** Notice that the list of message(s) that are considered suppressible in a conservative environment are identified by the full message identifier. In the list of the 'more aggressive' approach to message suppression, the messages are identified by the message prefix followed by an asterisk.

The five lists are:

- Conservative set of non-JES or system messages
- Aggressive set of non-JES or system messages
- Conservative set of JES2 messages
- Aggressive set of JES2 messages
- Conservative set of JES3 messages

The decision to designate a message eligible for automation processing is independent of the decision to suppress the message. You may want the message suppressed but not processed by automation. The two processes occur independently according to their individual specifications.

The following lists are concerned with suppression, not automation processing.

CONSERVATIVE LIST OF SUPPRESSIBLE NON-JES MESSAGES.

The following are examples. Review each message according to the environment in which it might be used. Make sure that you review each message and determine whether your installation should suppress it or not.

```
ARC0100I  SETSYS COMMAND COMPLETED                    HSM
ARC0200I  TRAP IN MODULE XXX
ARC0208I  TRAP FOR ERROR CODE XX
ARC0503I  ALLOCATION ERROR, RETURN CODE=XX
ARC0728I  VTOC FOR VOLUME XX COPIED TO DATA SET
ARC0734I  ACTION=MIGRATE FRVOL=XX TOVOL=XX TRACKS
CSV003I   MODULE NOT FOUND
CSV011I   FETCH FAILED
CSV300I   PROBABLE INVALID RECORD COUNT
DFS035I   BATCH INITIALIZATION COMPLETE
DFS092I   IMS/VS LOG TERMINATED
DFS627I   IMS/VS RESOURCE CLEANUP COMPLETED OR FAILED FOR JOB
DFS629I   IMS TCB ABEND IMS|SYS
DFS2207I  IMS/VS LOG(S) BLOCKSIZE=XXX,BUFNO=YYY
DFS2208I  XXXX LOGGING IN EFFECT ON IMS/VS ZZZZ
DFS2500I  *MDA00 DATABASE/DATASET ALLOCATED/UNALLOCATED
DSI090I   NCCF LOAD FAILED MSG DURING STARTUP
IAT4801   JOB JJJ EXPRESS CANCELLED BY INTERPRETER DSP
ICB084I   MSS TRACE STARTED
ICB086I   MSS TRACE X ENDED
ICB402I   VOLUME XXX NOT FOUND IN MSVC INVENTORY
ICB411I   UNABLE TO RESTORE BASE VOLUME XX RECORD
ICH70001I LAST ACCESS AT HH.MM.SS ON YY.DDD
ICH408I   LOGON WITH INVALID PASSWORD
ICH410I   RACF UNABLE TO BACKUP UPDATE ...
IEA848I   NO DUMP PRODUCED FOR THIS ABEND ...
IEA989I   SLIP TRAP MATCHED
IEA995I   SYMPTOM DUMP OUTPUT
IEC070I   (VSAM EOB ERROR)
IEC130I   DD STATEMENT MISSING
IEC141I   013-RC (open error)
IEC331I   ABEND MESSAGES
IEC161I   VSAM OPEN ERROR MESSAGES.
IEC705I   TAPE ON device IS label_type
IEC801I   LNA THRESHOLD TRANS= ...
IEC999I   IFG0TC0A...
IEE043I   SYSTEM LOG DATA SET HAS BEEN QUEUED
IEE400I   THESE MESSAGES CANCELLED - XX
IEF097I   USERID ASSIGNED
IEF125I   LOGGED ON
IEF126I   LOGGED OFF
IEF170I   (53 BYTES OF WTP TEXT)
IEF176I   WTR DDD WAITING FOR WORK
IEF188I   PROBLEM PROGRAM ATTRIBUTES ASSIGNED
IEF196I   ALLOCATION
IEF202I   STEP sss WAS NOT RUN BECAUSE OF cde
IEF236I   ALLOCATION FOR job
IEF237I   ALLOCATED TO
IEF287I   DSN DISP VOL SER NOS =
IEF288I   dsn SYSOUT
```

CONSERVATIVE LIST OF SUPPRESSIBLE NON-JES MESSAGES (continued)

```
IEF403I   job STARTED
IEF404I   job ENDED
IEF450I   JOB ABEND
IEF452I   JOB NOT RUN JCL ERROR
IEF453I   JOB FAILED JCL ERROR
IEF677I   WARNING MESSAGE(S) FOR JOB JOBNAME ISSUED
IEF710I   MSS MOUNT FAILED
IEF722I   JOB FAILED - SECURITY REASON
IEF861I   FOLLOWING RESERVED DATA SET NAMES UNAVAILABLE TO JOB
IEF863I   DSN=DSNAME
IKJ144I   UNDEFINED USER(S) XXX
IKJ572I   USER(S) XXX NOT LOGGED ON
IKJ605I   USER(S) XXX NOT LOGGED ON
IKJ606I   USERID ALREADY LOGGED ON
IKT100I   USERID CANCELLED DUE TO UNCONDITIONAL LOGOFF
IKT108I   USERID RECEIVE ERROR,RPLRTNCD=XX ETC.
IOS050I   CHANNEL DETECTED ERROR
IOS071I   (MIH message)
IST234I   I/O ERROR ON TERMINAL XXX
IST259I   INOP RECEIVED FOR NODENAME

IST521I   GBIND QUEUED FOR COS ETC
IST522I   ERNN ACTIVATION FAILED ...
IST523I   REASON =

IST530I   GBIND PENDING MESSAGE
IST532I   EVENT ID MESSAGE
IST619I   ID FAILED - RECOVERY IN PROGRESS
IST621I   RECOVERY SUCCESSFUL FOR NETWORK NODE
```

AGGRESSIVE LIST OF SUPPRESSIBLE NON-JES MESSAGES.

For a more aggressive approach to message suppression of non-JES messages, you might choose to add the following message prefixes to the preceding conservative list.

The following are examples. Review each message according to the environment in which it might be used. Make sure that you review each message and determine whether your installation should suppress it or not.

```
ADM*      ALL ADMPRINT MSGS
AHL*      ALL GTF MESSAGES
DFH1500   CICS INITIALIZATION MESSAGES
DFS0433   NUMBER OF BUFFERS FOR SUBPOOL SIZE NNN INCREASED
DFS0435   NUMBER OF BUFFERS INVALID ON CARD N
DFS0540   DFSZDC00 PROGRAM=, CKPTID=,
DFS0730   UNABLE TO OPEN/CLOSE DATASET WITH DDNAME ...
DFS391I   (Variable message from utility)
DFS826I   BLDL FAILED FOR FOLLOWING DBDs
DFS830I   BLDL FAILED FOR FOLLOWING PSBS;
DFS840I   INDEX ERROR ...
IAT7903   JOB JJJ HELD OUTPUT PURGED
IKF*      ALL COBOL MESSAGES
ERB*      RMF MESSAGES
IBM*      ALL PLI MESSAGES TO PROGRAMMER
IEC03*    B37-D37-E37 ETC. ABENDS
IEF251I   JOB CANCELLED
IFO*      ALL ASSEMBLER F MESSAGES
IFY*      ALL VS FORTRAN MESSAGES
VPW004*   VPW CLASS DEST STATUS         VIRTUAL PAPER WRITER
VPW012*   VPW IO ERROR(READ) INPUTDCB
VPW016*   VPW NO CATALOG POINTER
```

CONSERVATIVE LIST OF SUPPRESSIBLE JES2 MESSAGES.

To suppress JES2 messages through MPF processing, you must specify the "$" prefix. If you have more than one JES2 subsystem, however, the message suppressed by MPF will be suppressed for all JES2 subsystems.

The following are examples. Review each message according to the environment in which it might be used. Make sure that you review each message and determine whether your installation should suppress it or not.

```
$HASP001   R,TEXT VIA $DM
$HASP002   AUTOMATIC COMMANDS HALTED
$HASP100   ON TSORDR/INTRDR/ETC..... RDR
$HASP101   JOBNAME HELD
$HASP110   JOBNAME -- ILLEGAL JOB CARD
$HASP111   JOBNAME -- INVALID /*ROUTE CARD
$HASP112   JOBNAME -- INVALID /*JOBPARM CARD
$HASP113   JOBNAME -- INVALID /*OUTPUT CARD
$HASP114   JOBNAME -- INVALID EXECUTION MODE
$HASP115   JOBNAME -- INVALID /*NETACCT CARD
$HASP116   JOBNAME -- INVALID /*NOTIFY CARD
$HASP117   JOBNAME -- INVALID /*XMIT CARD
$HASP118   JOBNAME -- INVALID /*CONTROL STATEMENT
$HASP120   DEVNAME COMMAND (e.g. INTRDR $VS,'CMD')
$HASP125   SKIPPING FOR JOBCARD
$HASP160   DEVICE XXX INACTIVE
$HASP165   ENDED ...
$HASP200   XX STARTED ON LINEX
$HASP203   XX DISCONNECTED FROM LINEX
$HASP208   xxx SCHEDULED FOR yyy SNA,VTAM,...
$HASP210   SESSION zzzz LOGGED OFF LINElna
$HASP240   MESSAGES TO INACTIVE SPOOL MEMBER DISCARDED
$HASP244   XX INVALID COMMAND
$HASP249   NX $D CMD
$HASP250   JOB XXXXXXXX PURGED
$HASP301   JOBNAME - DUPLICATE JOB NAME - JOB DELAYED
$HASP309   INIT XX INACTIVE
$HASP310   XXX TERMINATED AT END OF MEMORY
$HASP395   JOBNAME ENDED
$HASP396   XXX TERMINATED
$HASP503   CONNECTION CONTROL...INVALID NODE NAME
$HASP520   XXX ON JOB TRANSMITTER
$HASP524   DEVICE INACTIVE
$HASP530   XXX JOBNAME ON DEVICE
$HASP532   XXX JOBNAME RESTARTED
$HASP534   DEVICE INACTIVE
$HASP540   XXX JOBNAME ON DEVICE
$HASP543   XXX JOBNAME DEVICE DELETED
```

AGGRESSIVE LIST OF SUPPRESSIBLE JES2 MESSAGES.

For a more aggressive approach to message suppression of JES2 messages, you might chose to add the following messages to the preceding conservative list.

The following are examples. Review each message according to the environment in which it might be used. Make sure that you review each message and determine whether your installation should suppress it or not.

```
$HASP094   I/O ERRORS ON LINE XX
$HASP150   OUTGRP ON device
$HASP308   JOBNAME - ESTIMATED TIME EXCEEDED
$HASP373   STARTING OF TASKS/JOBS/USERS
$HASP375   ESTIMATED LINES EXCEEDED
$HASP406   JOB WAS EXECUTING
```

CONSERVATIVE LIST OF SUPPRESSIBLE JES3 MESSAGES

With JES3, there is no particular category of messages that are generally suppressible. Therefore, the list contains only specific messages identified by complete message identifiers.

The following are examples. Review each message according to the environment in which it might be used. Make sure that you review each message and determine whether your installation should suppress it or not.

```
IAT1600    LINES EXCEEDED
IAT2000    JOB SELECTED
IAT2002    LSTOR=  ALLOC= ...
IAT2003    MPAINIT=  DI= ...
IAT2006    PREMATURE JOB TERM
IAT2007    GMS CONNECT - JOB JOBNO ...
IAT5200    JOB IN SETUP ON MAIN
IAT5916    MSS JOB JOBNO VOL...
IAT5918    MAIN JES3 V  (VERIFY DESCRIPTION)
IAT6101    JOB IS PRTY (job read in)
IAT6108    JOB (NOTIFY TEXT)
IAT6118    CARDS FLUSHED
IAT6160    JOB NET DJNET NOW ENTERING SYSTEM
IAT6201    JOB JOBNO=JJJ JOBS ...
IAT6306    JOB IS DSPNAME
IAT7001    JOB IS ON WRITER
IAT7007    JOB IS ON WRITER, PURGED
IAT7100    (INTERCOM cmd from dsp)
IAT7120    IO ERROR ON CONSOLE
IAT7310    NEW DJNET HAS COMPLETED
IAT7450    JOB PURGED
IAT7530    IO ERROR ON LINE
IAT9123    DATA RECEPTION ACTIVE ON LINE
IAT9124    JOB RECEPTION ACTIVE ON LINE
IAT9127    JOB IS FROM NODENAME
IAT9190    JOB IS BEING SENT ON LINE
IAT9191    JOB SENT TO NODE
```

Five Examples of MPFLSTxx Members

The contents of MPFLSTxx depend on the goals of your specific installation. The following five examples show possible contents of MPFLSTxx.

Example 1: Assume that you want to create a parmlib member named MPFLST7C to:

- Suppress some frequently issued JES2, MONITOR, and general messages that are of no interest to the operator or to automation processing.
- Set the color and highlighting attributes of messages to display eventual action messages in pink and non-action messages issued by problem programs in yellow.

The contents of MPFLST7C would be:

```
$HASP100
$HASP101
$HASP150
$HASP165
$HASP200
$HASP250
$HASP309
$HASP373
$HASP375
$HASP395
IEC130I
IEC705I
IEF125I
IEF126I
IEF165I
IEF170I
IEF236I
IEF237I
IEF403I
IEF404I
.MSGCOLR  EVETACTN(P,N),PPMSG(Y,N)
```

The SET MPF=7C command establishes this list of messages as those that are to be suppressed and ignored by an automation subsystem, and also sets the color and highlighting attributes for eventual action and non-action messages. The system uses the IBM-supplied defaults for all other message types or console fields.

Example 2: Assume that you want to create a parmlib member named MPFLSTDF to:

- Reset the color, highlighting, and intensity attributes to the IBM-supplied default values.
- Suppress, retain, and mark eligible for automation processing the same messages as specified in the previously active MPFLSTxx parmlib member.

The contents of MPFLSTDF would be:

```
.MSGCOLR DEFAULT
.MSGIDS NOCHANGE
```

Example 3: Assume that you want to create a parmlib member named MPFLSTRV to:

- Display all messages in reverse video (that is, the characters are to appear in black on a colored background).
- Display eventual action messages in pink and non-action messages issued by problem programs in yellow.
- Suppress, retain, and mark eligible for automation processing the same messages as specified in the previously active MPFLSTxx parmlib member.

The contents of MPFLSTRV would be:

```
.MSGCOLR URGATTN(R,R),IMEDACTN(W,R),EVETACTN(P,R)
.MSGCOLR GENMSG(G,R),PPMSG(Y,R),SELPEN(B,R)
.MSGCOLR INSTRERR(W,R),ENTRYARA(G,R),WARNLGEN(B,R)
.MSGCOLR WARNRGEN(B,R),WARNRURG(R,R),OOLCNTL(T,R)
.MSGCOLR OOLLABEL(T,R),OOLDATA(G,R)
.MSGIDS NOCHANGE
```

Example 4: Assume that you want to create a parmlib member named MPFLST18 to:

- Display all messages of the form IECxxxx and have control pass to an installation-supplied WTO/WTOR user exit routine each time the system prepares to issue such a message.
- Suppress certain IEFxxxx messages and have control pass to an installation-supplied WTO/WTOR user exit routine each time the system prepares to issue one of the messages.
- Allow action message IEE601E to be displayed at the operator console but not be retained by the action message retention facility.

The contents of MPFLST18 would be:

```
IEC*,SUP(NO),USEREXIT(usrexit1) /*usrexit1 handles IEC messages*/
IEF170I,SUP(YES),USEREXIT(usrexit2)
IEF236I,SUP(YES),USEREXIT(usrexit2)
IEF237I,SUP(YES),USEREXIT(usrexit2)
IEF403I,SUP(YES),USEREXIT(usrexit2)
IEE601E,SUP(NO),RETAIN(NO)          /*display only*/
```

Example 5: Assume that you want to create a parmlib member named MPFLST02 to:

- Specify that messages not identified in this member are not eligible for automation processing.
- Establish the default for specific messages so that they will be eligible for automation processing, suppressed, retained, and passed to a user exit. Override this default for one particular message and have it displayed.
- Establish the default for specific messages so that they will be eligible for automation processing, suppressed and retained. Display one specific message.
- Establish the default for specific messages so they will not be retained. They will be displayed and passed to a user exit for processing. The user exit must turn off automation processing and use the token to select a console to reroute the message to. A token value is not recognized before MVS/SP 2.2.0.

- The last .DEFAULT statement with no options specified sets the defaults for the subsequent list of messages to AUTO(NO) RETAIN(YES) and SUP(YES) and the messages will be passed to IEAVMXIT, if it exists.

The contents of MPFLST02 would be:

```
/*
/* DO NOT AUTOMATE THOSE MESSAGES NOT SPECIFIED IN MPFLST
/*
.NO_ENTRY,AUTO(NO)
/*
/* AUTOMATE CONSOLE BUFFERS AND USE MPF EXIT
/*
.DEFAULT,AUTO(YES),USEREXIT(COMMTASK)
IEA405E                          /*WTO BUFFER SHORTAGE - 80% FULL*/
IEA404A,SUP(NO)                   /*EVERE WTO BUFFER SHORTAGE - 100% FULL
                                    ALSO TELL OPER WHY JOBS IN WAIT*/
IEA406I                           /*TO BUFFER SHORTAGE RELIEVED*/
/*
/* AUTOMATE PRODUCTION JOB STATUS MESSAGES
.DEFAULT,AUTO(YES)
IEF402I,SUP(NO)                   /*JJJ FAILED IN ADDRESS SPACE X
                                     SYSTEM ABEND SXXX  REASON - RC
                                     NOTIFY OPER OF FAILURE /*
IEF403I                          /*JJJ-STARTED */
IEF404I                          /*JJJ-ENDED   */
/*
/* REROUTE THE TAPE MESSAGES
/*
/*   - AUTO TOKEN USED BY EXIT TO SELECT CONSOLE FOR MESSAGE
/*   - EXIT MUST TURN OFF AUTOMATION (IMPLIED BY TOKEN USE)
/*   - REROUTE EXIT WILL SEND MESSAGE TO PROPER CONSOLE
/*
.DEFAULT,RETAIN(NO),USEREXIT(REROUTE),AUTO(TAPEPOOL),SUP(NO)
IEF233*                     /*  M DDD,SER,LABEL,JJJ,SSS,DSN */
IEF234E                     /*  K/D/R DDD,SER PVT/PUB/DTR, JJJ,SSS,SPACE*/
IEC400A                     /*  M DDD,SER/DSN*/
IEC401A                     /*  F DDD,SER/DSN*/
IEC402D                     /*  F DDD,SER/DSN*/
IEC403A                     /*  M DDD,SER*/
IEC501*                     /*  M DDD,SER,LABEL,DENSITY,JJJ,SSS,DSN*/
IEC507D                     /*  E DDD,SER,JJJ,SSS,DSN*/
IEC509A                     /*  F DDD,SER,JJJ,SSS,DSN*/
/*
/*       RESET DEFAULT FOR OLD LIST
/*       (FROM A DIFFERENT MPFLSTxx MEMBER)
.DEFAULT
 IEC*
 IEF170I
 IEF236I
 IEF237I
 IEF403I
 IEE601E
```

Member Name: MVIKEY00

Use of the Member

MVIKEY00 supplies six parameters to the mass storage system communicator (MSSC). These parameters:

1. Give the location of messages for the space manager
2. Specify the window size for processing large sequential data sets
3. Specify the time interval for checking staging drive groups
4. Name the inventory and journal data sets
5. Specify the number of mounted mass storage volumes that the MSSC tracks
6. Specify the reason codes that cause the operator to either cancel or retry the job

Parameter in IEASYSxx (or specified by the operator):

None

Syntax Rules

The following rules apply when MVIKEY00 is updated with the IEBUPDTE utility:

- Use columns 1 through 71. Do not use columns 72-80, since these columns are ignored.
- Avoid embedded blanks.
- Separate consecutive parameters by a comma.
- Do not divide a parameter between consecutive records.
- Indicate continuation by a comma followed by one or more blanks after the last entry on a record.

Syntax Examples:

```
MSVCCAT=MSVICAT,MSSCSAMP=02,MSFMSG=02
MSVCCAT=USERCAT1,MSFMSG=(05,JOURNAL)
```

IBM-Supplied Defaults

The following defaults are automatically placed in MVIKEY00 at sysgen:

MSVCCAT=MSVCCAT,MSSCSAMP=03,MSFMSG=JOURNAL

Internal Parameters

The following parameters are optional and can be coded in any order.

Parameter	Meaning and Use	Value Range	Default
MSFMSG = { xx / JOURNAL / (xx,JOURNAL) }	Specifies where the space manager messages regarding the volume inventory status are to be written. xx identifies the route code of the alternate console where they will be written. JOURNAL specifies that the journal data set is to be used to record these messages. However, if the journal data set fills up, subsequent messages are lost unless they are also directed to a console.	01-16	JOURNAL
MSSCREA = { xx / (xx,xx,...) }	Specifies up to 15 reason codes. If a job fails with a reason code that matches the reason code specified for this parameter, the operator can retry or cancel the job.	01-FF	None
MSSCSAMP = { xx / 03 }	Specifies how often the MSSC will sample the load on each of the staging drive groups. The MSSC determines the least and most-used groups and passes this information to device allocation so device allocation can best choose a 3330V unit when it must mount a new MSS volume.	00-99 minutes	03
MSVCCAT = { xxxxxxxx / MSVCCAT }	Specifies the high qualifier name of the volume inventory and volume control journal data sets. The journal data set must be identified in the (master or user) VSAM Catalog as xxxxxxxx.MSVCJRNL and the inventory data set must be identified as xxxxxxxx.MSVI. If a VSAM user catalog is used, then the name of this user catalog must be xxxxxxxx. The inventory data set defaults to MSVCCAT.MSVI and the journal data set defaults to MSVCCAT.MSVCJRNL both cataloged in the VSAM master catalog or the VSAM user catalog MSVCCAT. MSVCCAT is automatically generated in the MSS logic at sysgen. **Note:** In a loosely coupled system, the journal and inventory data sets and the user catalog where they are cataloged must reside permanently on shared DASD. (For information on creating the inventory and journal data sets, see *Mass Storage System Extensions General Information* and *Mass Storage System Extensions Services: Guide*.)	1-8 alphameric characters**	MSVCCAT
TRKVOLS = xxx	Specifies the number of mounted mass storage volumes that the MSSC is to track. The MSSC uses this tracking information to eliminate unnecessary acquisitions of a cylinder.	0-999	68
WSIZE = pp	Specifies the number of pages that the Mass Storage System recognizes as a window. A window is the portion of a sequential data set that is staged for processing. A window is a multiple of a page (eight cylinders).	2-25	None

** First character must be *alphabetic*, #, @, or $. Following characters can be *alphameric*, #, @, $, hyphen, or 12/0 overpunch.

Member Name: PFKTABxx

Use of the Member

The PFKTABxx is an installation-created member of SYS1.PARMLIB. In this member, you define one or more program function key table(s) (PFK table). In another installation-created member of parmlib, CONSOLxx, you define the console configuration for the installation and specify which PFK member is to be used with that configuration. By using the CONSOLxx and PFKTABxx members together, the system can automatically initialize the console configuration with the related PFK table(s) during IPL. This processing reduces the number of operator responses, and eliminates the need to define manually the PF key settings.

Parameter in IEASYSxx (or entered by the operator):

None

Syntax Rules

The following rules apply to the creation of PFKTABxx by means of the IEBUPDTE utility:

- You may define multiple PFK tables in a member.
- Data must be contained in columns 1-71; the system ignores columns 72-80.
- Comments must begin with "/*" and end with "*/"
- The first statement type in a member must be PFKTAB TABLE(tablname).
- Code the statement type as the first data item on a record.
- For each PFK table, define each program function key only once.
- One or more blanks must follow the statement types; you must code at least one blank between the statement type and the keyword. (PFKTAB TABLE(tablname))
- Keyword values must be set off by parentheses. If you code multiple values on a keyword, separate the values with a blank or a comma.
- Do not use blanks, commas or comments in the middle of a keyword, between the keyword and the left parenthesis before the value, or in the middle of a value.
- A statement type continues to the next statement type in the member or until the end of the member. Therefore, there is no continuation character.

A Syntax Example:

```
PFKTAB TABLE(nnnnnnnn)

[          [ {CMD({"cccccc[;cccccc]..."}) } ] [         ] ]
[ PFK(xx)  [ {     {'cccccc[;cccccc]...'}  } ] [ CON({Y}) ] ]
[          [ {                             } ] [     {N}  ] ]
[          [ {KEY(kk[,kk]...)             } ]            ]
```

Using the Display Command

Once the PFKTABxx member has been invoked through the SET PFK command, you may use the DISPLAY command to view the contents of that specific PFKTABxx member. See *System Commands* for the correct syntax.

IBM-Supplied Defaults

The IBM-supplied default member of SYS1.PARMLIB is PFKTAB00.

Internal Parameters

Statement	Keyword(Value)	Meaning and Use
PFKTAB	TABLE (tablname)	PFKTAB indicates that a new PFK table is being defined. PFKTAB must be the first definition in a PFKTABxx member. (*tablname*) indicates the name associated with this PFK table. tablname must be 1-8 alphanumeric characters. The value you specify for tablname here is the value you specify for PFKTAB(tablname) in the CONSOLxx parmlib member.
	PFK(xx)	Indicates the program function key that is being defined. (*xx*) is a decimal value from 1 through 24. Each xx value must be unique within a table.
	CMD { ('command[;command]...') / ('command[;command]...') }	CMD indicates that the PFK is to have a command or commands associated with it. *command* specifies the command. Multiple commands on a CMD line must be separated with a semi-colon. The maximum length of the commands including the single quotes is 126 characters. If a command contains a single quote, surround the command with double quotes. If the command contain double quotes, surround the command with single quotes. If a command contains an underscore, define the key as conversational CON(Y). When the system displays the command, the cursor will be under the character immediately to the right of the underscore. The system will not display the underscore. See *System Commands* for details. JES3 commands must have an asterisk (*) for a prefix or an alternate prefix as specified on the CONSTD statement of the JES3 initialization deck.
	KEY { (xx) / (xx,xx...) }	KEY indicates that the PFK being defined is to be associated with another key or a list of PF keys. *xx* indicates the PFK to be processed when the PFK being defined is pressed. *xx* is a decimal number between 1-24 and can not be the same value as on the PFK(xx). For example, do not define PFK(10) KEY(10). You may code a maximum of 62 keys in a key list.
	CON { (Y) / (N) }	Specifies whether the PFK being defined is to be conversational or nonconversational. If the PFK is conversational, CON(Y), the system will display the command so you can modify it before executing it. If the PFK being defined is associated with a list of keys, and the PFK is conversational, the system displays the command associated with each key in the key list so you can make modifications. If a PFK is nonconversational, the system automatically executes the command when the PFK is pressed. If you do not specify CON, the default is nonconversational; the command will not be displayed before executing. **Note:** If a command contains an underscore, you must specify CON(Y) with it.

Member Name: SCHEDxx

Use of the Member

The SCHEDxx parmlib member allows the installation to have central control over which eligible device table the system is to use, the size of the master trace table, the abend codes that are to be eligible for automatic restart, and which programs requiring special attributes are to be included in the program properties table (PPT).

There are five statement types that you can use in the SCHEDxx member:

EDT
identifies which eligible device table the system is to use.

MT
defines the size of the master trace table if one exists.

PPT
allows the installation to specify a list of programs that require special attributes or to change the attributes of the IBM-supplied defaults.

For more information on the program properties table see *System Modifications*.

RESTART
allows you to add user abend codes to the list of abend codes that are eligible for automatic restart.

NORESTART
allows you to delete system and user abend codes from the list of abend codes that are eligible for automatic restart.

RESTART and NORESTART allows the installation to customize the list of abend codes that are eligible for automatic restart. This list is created by merging the user abend codes specified on the RESTART statement with the list of IBM-supplied system abend codes. A user abend code cannot have the same number as a system abend code. The system will ignore a user abend code that is a duplicate of a system abend code. The IBM-supplied system abend codes are:

001	20A	422	813
031	213	513	837
033	214	514	906
03A	217	613	913
0A3	2F3	614	926
0B0	313	626	937
0F3	314	637	A14
100	317	700	B14
106	32D	714	B37
113	413	717	C13
117	414	737	E1F
137	417	806	E37

Parameter in IEASYSxx (or specified by the operator):

$$\text{SCH} = \left\{ \begin{array}{l} \text{aa,} \\ \text{(aa)} \\ \text{(aa,L)} \\ \text{(aa,bb...,L)} \end{array} \right\}$$

The two alphameric characters, (aa, bb, and so forth) are appended to SCHED to form the name of the SCHEDxx member(s).

Note: The contents of SCHEDxx appears on the operator's console if the (,L) option is specified with either the SCH= keyword in the IEASYSxx parmlib member or in response to the 'SPECIFY SYSTEM PARAMETERS' prompt.

Syntax Rules

The following rules apply to the creation of SCHEDxx by means of the IEBUPDTE utility:

- Use columns 1 through 71. Do not use columns 72-80, because the system ignores these columns.
- Comments may be appear in columns 1-71 and must begin with "/*" and end with "*/"
- A statement type consists of 1-10 characters.
- A statement must begin with a valid statement type followed by at least one blank.
- A statement ends with the beginning of the next valid statement type or EOF.
- A statement can be continued even though there is no explicit continuation character.
- Multiple statement types are accepted.
- Operands must be 60 characters or less and may not span multiple records.
- Operands must be separated by valid delimiters. Valid delimiters are a comma, a blank, or column 71. If the operand contains parenthesis, then the right parenthesis is accepted as a valid delimiter.
- Multiple occurrences of a delimiter are accepted but treated as one.

IBM-Supplied Defaults

None

Internal Statement Types

The SCHED statements are described as follows:

Statement		Meaning and Use	Value Range	Default
EDT	ID(xx)	Specifies the eligible device table (IEFEDTxx) that the system is to load from SYS1.NUCLEUS. xx is two alphameric characters. The MVS Configuration Program builds the IEFEDTxx. **Notes:** 1. If you specify an IOCONFIG id on the SYSCTL frame and you do not specify an EDT id in SCHEDxx, the default for the EDT id is the same as the *IOCONFIG id.* 2. If you specify neither an IOCONFIG id nor an EDT id, the default for both is *00.* 3. If you do not specify an IOCONFIG id but you do have an EDT id, the EDT will be the one specified in the SCHEDxx member.	A-Z 0-9	00 or the IOCONFIG id
MT	SIZE { nnnK / NONE / 24K }	A master trace table is to be created by master scheduler initialization. This table is used by the TRACE command 'SIZE nnnK' specifies the size of the table. nnn can be any value from 16 to 999. If you specify MT=NONE, no master trace table is created. If there are multiple MT statements, the system processes the first entry and ignores any duplicate statements.		24K
NORESTART	CODES(code,code...)	Specifies which system and user abend codes are to be deleted from the table of abend codes that are eligible for automatic restart.	0-FFF (Hex)	
RESTART	CODES(code,code...)	Specifies the user abend codes that are to be eligible for automatic restart. These are to be added to the system abend codes that are supplied by IBM.	0-FFF (Hex)	
PPT		Allows the installation to specify a list of programs that require special attributes. Entries for these programs are created in the Program Properties Table (PPT). For more information on the PPT and the IBM-supplied defaults, see *System Modifications.*		
	PGMNAME(name)	PGMNAME(name) identifies by name the program that requires special attributes. It must consist of an alphabetic or national character followed by 0 to 7 alphanumeric or national characters. PGMNAME(name)is required on the PPT statement.		NONE

Statement		Meaning and Use	Value Range	Default
	CANCEL / NOCANCEL	The program specified on PGMNAME can be cancelled (CANCEL) or can not be cancelled (NOCANCEL).		CANCEL
	KEY(n)	The program specified on PGMNAME is to have the protection key (n) assigned to it. The range of values for n is 0 through 15.	0-15	Key 8
	SWAP / NOSWAP	The program specified on PGMNAME is swappable (SWAP) or non-swappable (NOSWAP).		SWAP
	PRIV / NOPRIV	The program specified on PGMNAME is privileged (PRIV) or not privileged (NOPRIV).		NOPRIV
	DSI / NODSI	The program specified on PGMNAME requires data set integrity (DSI) or does not require data set integrity (NODSI). For NODSI, the job must be a one-step job.		DSI
	PASS / NOPASS	The program specified on PGMNAME is allowed to bypass password protection (NOPASS) or is not allowed to bypass password protection (PASS).		PASS
	SYST / NOSYST	The program specified on PGMNAME is a system task and is not timed (SYST) or is not a system task and is to be timed (NOSYST).		NOSYST
	AFF(a,[b,...]) / AFF(NONE)	The program specified on PGMNAME must execute on a specific processor. a, b, ...identifies the processor number(s) that the program must execute on. If you omit AFF or specify AFF(NONE) the program has no processor affinity. Do not code AFF for a program that contains vector instructions.	0-15	AFF(NONE)
	SPREF	The program specified on PGMNAME must have all private area short-term fixed pages assigned to preferred storage frames.		None
	LPREF	The program specified on PGMNAME must have all private area long-term fixed pages assigned to preferred storage frames.		None
	NOPREF	The program specified on PGMNAME does not need to have all private area short-term fixed pages assigned to preferred storage frames.		None

Example of Member

The following is an EXAMPLE of how to use the SCHEDxx member of SYS1.PARMLIB. The data represented on each statement is not necessarily the IBM-supplied value. An installation may use this example and modify it according to its needs.

```
/*********************************************************************/

MT   SIZE(24K)              /*Default size of the master trace table*/
EDT  ID(01)                 /*The eligible device table defined in  */
                            /*IEFEDT01 will be loaded into storage  */

/*********************************************************************/
/*           Program Properties table addition                       */
/*                for a CICS program.                                */
/*                                                                   */
/*    The following defaults will be taken for this entry:           */
/*     No affinity to a particular processor      (AFF(NONE))        */
/*     Cancellable                                (CANCEL)           */
/*     Requires data set integrity                (DSI)              */
/*     Not a privileged job                       (NOPRIV)           */
/*     Not a system task                          (NOSYST)           */
/*********************************************************************/

PPT     PGMNAME(DFHSIP)         /*Add program name DFHSIP to the PPT */
        KEY(8)                   /*Protection key 8                  */
        NOSWAP                   /*Non-swappable                     */
        NOPASS                   /*Bypass password protection        */
        NOPREF                   /*No preferred storage required     */

/*********************************************************************/
/*           Program Properties table addition                       */
/*            for Cryptographic Unit Support                         */
/*                                                                   */
/*    The following defaults will be taken for this entry:           */
/*     No affinity to a particular processor      (AFF(NONE))        */
/*     Cancellable                                (CANCEL)           */
/*     Requires date integrity                    (DSI)              */
/*     Requires password protection               (PASS)             */
/*     Swappable                                  (SWAP)             */
/*********************************************************************/

PPT     PGMNAME(ICUMKM11)       /*Add program name ICUMKM11 to the PPT*/
        KEY(1)                  /*Protection key 1                   */
        PRIV                    /*Privileged                         */
        SYST                    /*Syst task, not timed               */

/*********************************************************************/
/*           Program Properties table replacement                    */
/*            for IBM-supplied PPT entry for TCAM                    */
/*                                                                   */
/*    The following defaults will be taken for this entry:           */
/*     Not a privileged job                       (NOPRIV)           */
/*     Not a system task                          (NOSYST)           */
/*********************************************************************/
```

```
PPT     PGMNAME(IEDQTCAM)     /*Add program name IEDQTCAM to the PPT  */
        AFF(1)                /*This job must execute on processor 1  */
        CANCEL                /*Job is cancellable                    */
        DSI                   /*Requires data set protection          */
        KEY(6)                /*Protection key 6                      */
        NOPREF                /*Private area short-term fixed pages   */
                              /*Need not be assigned to preferred     */
                              /*storage frames                        */
        NOSWAP                /*Non-swappable                         */
        PASS                  /*Requires password protection          */
/*********************************************************************/
/*            ADDITION to the Table of Codes Eligible                */
/*              For Automatic Restart                                */
/*    Note:  Completion codes may only be added to the user-defined  */
/*           portion of the table.                                   */
/*           The following codes will be added to the                */
/*           Table of Codes Eligible for Automatic Restart:          */
/*                CCC    DDD   EEE                                   */
/*********************************************************************/

RESTART CODES(CCC,DDD,EEE)    /*Add these codes to the Table of Codes*/
                              /*Eligible for Automatic Restart       */

/*********************************************************************/
/*                                                                   */
/*            DELETION from the Table of Codes Eligible              */
/*              for Automatic Restart                                */
/*    Note:  Completion codes may be deleted from either the         */
/*           user-defined portion of the table or from the           */
/*           system-defined portion of the table.                    */
/*                                                                   */
/*    The following codes will be deleted from the Table of Codes    */
/*          Eligible for Automatic Restart:                          */
/*                                                                   */
/*                  213        B37                                   */
/*                                                                   */
/*********************************************************************/

NORESTART CODES(213,B37)  /*Delete these codes from the Table of Codes  /*
                          /*Eligible for Automatic Restart              /*
```

Member Name: SMFPRMxx

Use of the Member

The SMF parameters allow you to:

- Specify the job wait time limit
- Identify the system on which SMF is active
- Select the record types and subtypes SMF is to write
- Allow the operator to change the SMF parameters established at IPL.
- Specify whether SMF is to invoke user-written exits.

Note: The SMF data sets must be cataloged on DASD. If they are not, SMF data is not recorded. For a full description of SMF requirements and use, see *SPL: SMF*.

You can specify SMF parameters in several ways:

- Before the first IPL of a newly generated system by creating an SMFPRMxx parmlib member.
- At each initialization of SMF by entering the parameters at the console.
- During SMF execution by using the SET SMF command to specify a different SMFPRMxx parmlib member or by using the SETSMF command to replace one or more previously defined SMF parameters.

Using the SET Command

The SET operator command can be used to modify the SMF recording options dynamically by specifying which SMFPRMxx parmlib member is to be used. Also, if SMF terminates, the SET SMF command can be used to restart SMF. The format of the command is:

```
SET SMF=xx
```

where xx is the two-character identifier of the SMFPRMxx member of SYS1.PARMLIB that contains the SMF options.

Using the SET command, the installation can replace all the existing SMF options. For example, an installation can activate SMF recording after an IPL in which NOACTIVE is specified by using the SET command and choosing the parmlib member that contains the ACTIVE option. In addition, the installation can use the SET command to reactivate SMF recording after an I/O error has terminated recording; however, the installation should define a new data set or correct the cause of the I/O error before reactivating SMF recording.

Notes:

1. To avoid installation exit communication problems, the installation should terminate all address spaces except the master scheduler address space, the system address spaces (such as PCAUTH and ALLOCAS), and the job entry subsystem before issuing a SET command that changes the EXITS keyword.
2. The SET and SETSMF commands cannot be used to change the SID parameter.
3. The new values for STATUS or MAXDORM do not take effect until the old ones, if any, expire.
4. SET SMF, SETSMF, and DISPLAY SMF commands cannot run simultaneously. One command waits for another to complete before starting.

5. For each IPL, a maximum of eight subsystems can be defined to SMF (via the SUBSYS parameter). This is a combined total of those specified at IPL and subsequent SET commands. If the maximum is reached, no new subsystems may be added. Those subsystems previously specified can be given different options.
6. Recording data set switching does not take place at SET time unless it is necessary. For example, if the current active data set is not included in the new options, the first empty data set in the new data set list becomes the active recording data set.
7. If recording is not active at SET SMF time, the first non-full data set is used as the recording data set.

Using the SETSMF Command

In contrast to the SET SMF command, which allows an installation to specify a different SMFPRMxx parmlib member or to restart SMF, the SETSMF operator command allows an installation to:

- Add a SUBPARM parameter value to those SMFPRMxx parameter values already set for this IPL.
- Replace SMFPRMxx parameter values (except ACTIVE, PROMPT, SID, and EXIT) with new ones for this IPL.

If the SMFPRMxx parameter values set for this IPL include NOPROMPT, then the operator cannot use the SETSMF command to modify any of these values or to add the SUBPARM parameter value. The format of the SETSMF command is:

```
{SETSMF}  parameter (value),...
{SS    }
```

The parameters are:

parameter
specifies any SMF parmlib parameter except ACTIVE, PROMPT, SID, or EXITS.

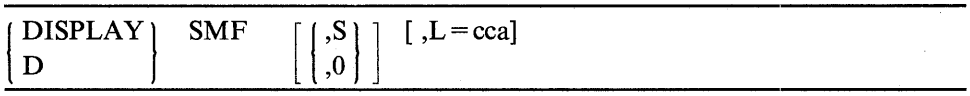
value
specifies the new value for the specified parameter.

Notes:

1. More than one parameter can be changed as long as the length of the command does not exceed 124 characters.
2. Both the SUBSYS and SUBPARM specifications can be changed on the same SETSMF command as long as the subsystem name is the same.

Using the DISPLAY Command

The DISPLAY (D) operator command can be used to display the status of the SMF data sets or the current SMFPRMxx options, to the operator console. The format is:



The parameters are explained below:

,S
specifies that the status of the SMF data sets is to be displayed.

,O
specifies that the current SMFPRMxx options are to be displayed.

,L=cca
specifies the console(cc) and the display area(a) where the action specified by S or O is to take place.

Note: SET SMF and DISPLAY SMF commands cannot run simultaneously. One waits for the other to complete before starting.

Parameter in IEASYSxx (or specified by the operator):

SMF=xx

The two alphameric characters, represented by xx, are appended to SMFPRM to identify the SMFPRMxx member of parmlib. If the parameter is not specified either in IEASYSxx or by the operator, the system uses the default member SMFPRM00.

Syntax Rules

The following rules apply to the creation of SMFPRMxx by means of the IEBUPDTE utility:

- Use columns 1 through 71. Do not use columns 72-80, since these columns are ignored.
- Avoid embedded blanks.
- Enter each parameter in the format: keyword (value).
- Indicate continuation by placing a comma after the last entry on a record, followed by a blank before column 72.

Syntax Example:

```
SID(3090),ACTIVE,
DSNAME(SYS1.MANA,SYS1.MANB,SYS1.MANC),
JWT(0030),SYS(TYPE(00:120),NOEXITS,
INTERVAL(004000),DETAIL)
```

IBM-Supplied Defaults

The default member, SMFPRM00, contains the following parameters:

```
ACTIVE                          /*ACTIVE SMF RECORDING*/
DSNAME(SYS1.MANX,SYS1.MANY)     /*TWO DATA SETS MANX AND MANY*/
PROMPT(ALL)                     /*PROMPT THE OPERATOR FOR OPTIONS*/
REC(PERM)                       /*TYPE 17 PERM RECORDS ONLY*/
MAXDORM(3000)                   /*WRITE AN IDLE BUFFER AFTER
                                  30 MIN*/
STATUS(010000)                  /*WRITE SMF STATS AFTER 1 HOUR*/
JWT(0010)                       /*522 AFTER 10 MINUTES*/
SID(3090)                       /*SYSTEM ID IS 3090*/
LISTDSN                         /*LIST DATA SET STATUS AT IPL*/
SYS(TYPE(0:255),EXITS(IEFU83,IEFU84,IEFACTRT,IEFUJV,IEFUSI,
       IEFUJP,IEFUSO,IEFUJI,IEFUTL,IEFU29),NOINTERVAL,NODETAIL)
/* WRITE ALL RECORDS AS THE SYSTEM DEFAULT, TAKE ALL KNOWN
   EXITS. THERE ARE NO DEFAULT INTERVAL RECORDS WRITTEN.
   ONLY SUMMARY TYPE 32 RECORDS ARE WRITTEN FOR TSO.*/
SUBSYS(STC,EXITS(IEFU29,IEFU83,IEFU84,IEFUJP,IEFUSO))
/* WRITE ALL RECORDS AS BY SYSTEM DEFAULT, TAKE ONLY FIVE
   EXITS, NOTE:  IEFU29 EXECUTES IN THE MASTER ASID WHICH
   IS A STC ADDRESS SPACE SO IEFU29 MUST BE ON FOR STC.
   USE ALL OTHER SYS PARAMETERS AS A DEFAULT*/
```

You should modify this list according to your system requirements. You may place alternate values, plus additional values, in one or more alternate SMFPRMxx lists.

Internal Parameters

The SMF parameters are described as follows:

Parameter	Meaning and Use	Value Range	Default
ACTIVE NOACTIVE	Specifies whether or not SMF recording is to be active.	N/A	ACTIVE
DSNAME (data set)	Specifies a list of up to 36 VSAM data sets that are to be used for SMF recording. Each data set must be named SYS1.MANn, where n can be any alphameric character. The first data set specified is the primary SMF recording data set. All other specified data sets are secondary data sets.	See explanation	DSNAME (SYS1.MANX, SYS1.MANY)
LISTDSN NOLISTDSN	Specifies whether the system is to generate SMF data set status messages to the operator at IPL or SET SMF time. The messages contain the following information for each data set used for recording: • data set name • data set status – active – alternate – dump required • data set size (in number of 4096-byte blocks) • percentage full	N/A	LISTDSN
SID { (xxxx) | (xxxx,ser#[,ser#]) }	Specifies the system identifier to be used in all SMF records, where xxxx can be any four alphameric and/or special characters. If you want to use one SMFPRMxx parmlib member for your entire installation, specify a SID parameter for each system and use the serial number subparameter on each SID parameter. SMF selects the SID with the serial number(s) that match the processor serial number(s) at IPL. The xxxx value associated with the serial number(s) will be the system identifier placed in each SMF record. For example, if SID(SYSA,006204) and SID(SYSB,006204,206204) are specified in the same parmlib member, and processor 006204 was IPLed as a uniprocessor, SID(SYSA,006204) applies. If the processor was IPLed as a multiprocessor, SID(SYSB,006204,206204) applies. **Notes:** 1. The SID parameter cannot be changed by a SET SMF or a SETSMF command. 2. Note that it is possible for an installation to have more than one processor (such as a 3090) with the same serial number. In such cases, use a separate SMFPRMxx parmlib member for each processor. Otherwise, SMF will select the system identifier that contains the first occurrence of the matching serial number, regardless of the value specified for xxxx.	See explanation	SID(3090) The SID assigned is the four-digit processor model number taken from the PCCA control block.

Parameter	Meaning and Use	Value Range	Default
REC({ ALL) | PERM }	Specifies whether information for record type 17 (scratch data set status) is to be collected for temporary data sets. PERM specifies that record type 17 is to be written only for non-temporary data sets. ALL specifies that record type 17 is to be written for both temporary and non-temporary data sets.	N/A	REC (PERM)
MAXDORM (mmss) NOMAXDORM	Specifies the amount of real time that SMF allows data to remain in an SMF buffer before it is written to a recording data set, where mm is real time in minutes and ss is seconds. NOMAXDORM specifies that the data remains in the buffer until the buffer is full.	0001-5959	MAXDORM (3000) 30 minutes
STATUS (hhmmss) NOSTATUS	Specifies the amount of real time between creations of record type 23 (SMF statistics) where hh is the hours, mm is the minutes, and ss is the seconds.	000001-240000	STATUS (010000) 1 hour
JWT (hhmm)	Specifies the maximum amount of time that a job or TSO user is allowed to wait continuously, where hh is the amount of real time in hours and mm is in minutes. When the specified time limit has expired, the time limit exit, IEFUTL, is entered (if active). IEFUTL cancels the job and automatically logs off the TSO user. **Note:** If TIME = 1440 is coded on the JOB or EXEC JCL statement, IEFUTL is not invoked for that job.	0001-2400	JWT (0010) 10 minutes
PROMPT (option) NOPROMPT	Specifies whether the selected SMF parameters are to be displayed on the system console at IPL time. The operator can be prompted for a reason for the IPL or to modify the parmlib parameters. The options are as follows: • IPLR specifies that the operator is to supply a reason for the IPL. • LIST specifies that the operator can modify the SMF parameters. • ALL specifies that the operator is prompted for the IPL reason and can modify the SMF parameters. PROMPT specifies that the parameters are not listed and the operator is not prompted unless there is a syntax error in the parmlib member.	N/A	PROMPT (ALL)

Parameter	Meaning and Use	Value Range	Default
SYS (options)	Specifies the global recording options for the entire system. The options include the record types and subtypes to be collected, the real time intervals between recording, and the valid SMF exits. If the same option is specified more than once, the first valid specification read from the parmlib or the last one specified in an operator reply is used. The options are as follows:	See each option	See each option
	TYPE NOTYPE ({ aa,bb(cc) / aa,bb:zz / aa,dd(cc:yy),... / aa,bb(cc,...) }) TYPE specifies the record types and subtypes that SMF is to collect. aa, bb, dd, and zz are the decimal notations for each record type. cc and yy are the decimal notations for the record subtypes. A colon indicates the range of record types (bb through zz) to be recorded or the range of subtypes (cc through yy for record type dd) to be recorded. Subtypes are valid on record types 0-127. NOTYPE specifies that SMF is to collect all record types and subtypes ***except*** those specified. aa, bb, and zz are the decimal notations for each record type. cc and yy are the decimal notations for each subtype. A colon indicates the range of record types (bb through zz) or the range of subtypes (cc through yy for record dd) that are ***not*** to be recorded.	0-255 (record types) 0-127 (subtypes)	TYPE (0:255) (all types and subtypes)
	NOINTERVAL INTERVAL (hhmmss) NOINTERVAL specifies that no checkpoints are taken. NOINTERVAL is the default. INTERVAL specifies the amount of real time between each checkpoint, where hh is the hours, mm is the minutes, and ss is the seconds. At each checkpoint, a type 30 record if specified, is written. For TSO users, a type 32 record can also be written. The INTERVAL option allows the user to preserve accounting data for long-running jobs or TSO sessions. Because accounting data is recorded for each job or task each time the interval expires, the data is not completely lost in the event of a system failure. However, using the interval option causes some system overhead. For this reason, caution should be used when specifying this value.	000001-240000	NOINTERVAL

Parameter	Meaning and Use	Value Range	Default
	EXITS (exit name, exit name,...) NOEXITS EXITS specifies which SMF exits are to be invoked. A maximum of 15 exits are allowed; if an exit is not specified, then it is not invoked. If this system parameter is not specified, all SMF system exits are allowed. NOEXITS specifies that SMF exits are not invoked.	N/A	See explanation
	NODETAIL DETAIL Specifies the level of data collection. When DETAIL is specified for TSO, type 32 records contain the total CPU time under TCBs and SRBs and the total number of TGETs, TPUTs, EXCPs, and transactions associated with the command. When DETAIL is specified for STC, type 30 records (subtypes 4 and 5) contain all the EXCP sections for the step or job.		NODETAIL
SUBPARM (name(parameter))	Specifies the information to be passed to a specific subsystem where: *name* specifies a one to four character subsystem name. The first character must be alphabetic or #, @, or $, and the remaining characters can be either alphameric or national. *parameters* specifies a 1 to 60 character information string to be passed to the subsystem specified in *name*. SMF does not check the validity of the information string. The inner set of parentheses marks the beginning and the end of the information string.	N/A	None
SUBSYS (name,options)	Specifies what data is to be collected and recorded for a specific subsystem, where name specifies a one to four character subsystem name. The first character must be alphabetic or #, @, or $, and the remaining characters can be either alphameric or national. The options are the same as the options that can be specified with the SYS parameter including subtype specifications. If you omit SUBSYS, SMF uses the values on the corresponding SYS option. The name is not validity checked. Data can be recorded for up to eight subsystems in any IPL, including those specified at IPL and through subsequent SET commands. When the limit is reached, no additional subsystems can be added. The two SMF-defined subsystems are STC and TSO. Work started from the operator console is associated with the STC subsystem. Logged on TSO users are assigned to the TSO subsystem. Batch jobs are assigned to the job entry subsystem that presented the work to the system.		

Member Name: TSOKEY00

Use of the Member

TSOKEY00 contains TSO/VTAM time-sharing parameters.

Starting TSO/VTAM time sharing activates the terminal control address space (TCAS). The function of TCAS is to accept TSO/VTAM logon requests and to create an address space for each TSO user. TCAS builds a TCAS table (TCAST) and inserts the parameter values into it. The VTAM terminal I/O coordinator (VTIOC), which is the interface between TSO and VTAM, uses these values to control the time-sharing buffers, the maximum number of users, and other operational variables.

TSOKEY00 or an alternate member name may be specified by using the MEMBER operand or the keyword operand (overriding keyword MBR) of the operator START command, or by coding it into the PARMLIB DD statement of the cataloged procedure evoked by the START command. If a member name is not specified but a parmlib name is specified (on the PARMLIB DD statement), the system defaults to TSOKEY00. If neither a member name nor a parmlib name is specified, default values in the TCAS program are used.

If, during TCAS initialization, an I/O error occurs after some of the values have been read from parmlib, the default values in the TCAS program supply the remaining parameter values for the TCAS table.

If the specified value for a parameter does not fall within the value range, the default value is used.

Parameter in IEASYSxx (or specified by the operator):

None

Syntax Rules

The following rules apply to the creation of the TSO/VTAM time-sharing parmlib member by means of the IEBUPDTE utility:

- For each record, columns 1 through 71 are valid for data. Columns 72 through 80 are ignored.
- Data may be continued from one record to another by ending a record with a comma and blank in contiguous data columns, then inserting data in any column of the data area on the next record.
- A parameter must be complete in a record. It may not cross record boundaries. It may not be repeated.
- A comma must separate adjacent keywords.
- Invalid or misspelled parameters are ignored. Default values are substituted, and are listed on the device specified by the PRINTOUT DD statement of the procedure used to start TSO/VTAM time sharing, or on the device specified by the device name operand of the operator START command.

IBM-Supplied Defaults

The default values are internal constants in the TCAS program.

USERMAX=40,RECONLIM=3,BUFRSIZE=132,HIBFREXT=48000,
LOBFREXT=24000,CHNLEN=4,SCRSIZE=480,ACBPW=password
(optional - no default),MODE=NOBREAK,MODESW=NO,
RCFBDUMP=xxyyz (optional - no default),CONFTXT=YES

Internal Parameters

The TSO/VTAM time sharing parameters are as follows:

Parameter	Meaning and Use	Value Range	Default Value
USERMAX	Specifies the maximum number of users that may be logged on to the TSO/VTAM time sharing system at one time. (Note that because VTAM considers each user to be an application program, and because there must be an APPL definition statement that defines each application program to VTAM during VTAM network definition, this value may not exceed the number of APPL definition statements.)	0-maximum number of address spaces in the system	40
RECONLIM	Specifies the time limit in minutes within which a user may reconnect after his line has been disconnected (that is, the amount of time an address space remains active in the event of terminal disconnection).	0-32,767	3
BUFRSIZE	Specifies the size in bytes of a VTIOC buffer. Input and output data smaller than the BUFRSIZE value uses cells equal to the BUFRSIZE value. Input and output data larger than the BUFRSIZE value is assigned to dynamically allocated buffers (allocated by the GETMAIN macro) equal to the data size.	4-3,072	132
HIBFREXT	Specifies the maximum amount (in bytes) of virtual storage that can be dynamically allocated for output data. When this value is reached, no more output requests are honored until LOBFREXT is reached. (Input requests are rejected only when no virtual storage is available in the address space.)	Maximum size TPUT used at the installation -maximum amount of available virtual storage	48,000
LOBFREXT	Specifies the minimum number of virtual storage bytes that can be dynamically allocated for output data. When this value is reached, tasks that are suspended for lack of output buffers are marked as dispatchable.	0-(HIBFREXT-1)	24,000
CHNLEN	Specifies for output the number of request units (RUs) in each RU chain for IBM 3767 and IBM 3770 terminals. This value determines the maximum size of a chain and, therefore, the maximum amount of data that can be sent to the terminal in one chain (CHNLEN x 256 = data transmitted). This value also determines the maximum amount of data purged when the CANCEL key is pressed to stop printing of outbound (host-to-terminal) data. The CHNLEN value is associated with the values specified on the PACING operand of the LU macro during network control program definition, and on the VPACING operand of the LU or PU macro during VTAM definition. Use the CHNLEN default value (4) if the PACING and VPACING default values are used. Otherwise, calculate CHNLEN by using the formula *(2pn-2pm) + (2vn-2vm)*, where *pn* and *pm* are the PACING *n* and *m* values, and *vn* and *vm* are the VPACING *n* and *m* values. If more than one network control program is used with TSO/VTAM time sharing, the smallest value found when calculating the CHNLEN value should be used. Otherwise, the network control program could enter slowdown mode.	1-10	4

Parameter	Meaning and Use	Value Range	Default Value
SCRSIZE	Specifies the default screen size of IBM 3270 systems network architecture (SNA) terminals. (Use the FEATUR2 operand of the LOCAL or TERMINAL macros to specify non-SNA IBM 3270 screen sizes.) This value affects only those terminals whose sizes were not specified by using the PSERVIC parameter of the MODEENT macro during VTAM definition. If an installation has IBM 3270 SNA terminals of only one screen size, SCRSIZE should specify that size. If the installation has IBM 3270 SNA terminals of both screen sizes, SCRSIZE may be used to define the more common size, and the PSERVIC parameter may be used to define the less common size.	screen size of the terminal	480
ACBPW	Specifies the optional password associated with all TSO/VTAM ACBs. If a password is specified on the PRTCT parameter of VTAM's APPL definition statement, ACBPW must be specified. To prevent unauthorized disclosure of the ACB password when the CBPW keyword is specified, do the following: **Notes:** 1. Place the TSOKEY00 member in a password protected data set that is compatible with SYS1.PARMLIB (for example SYS1.VTAMLST). 2. Modify the parmlib DD statement in the TSO/VTAM start procedure to refer to the password protected data set. For example: `//PARMLIB DD DSN=SYS1.VTAMLST(&MBR),DISP=SHR,FREE=CLOSE`	Any 1 to 8 EBCDIC characters	None
MODE	Specifies how 3276 and 3278 terminals are to be supported for TSO/VTAM. BREAK indicates that the terminal keyboard is unlocked whenever possible and does not necessarily correspond to the issuance of a TGET. NOBREAK indicates that the terminal keyboard is locked except when a TGET is issued. The BREAKIN option of TPUT applies only when BREAK is specified.	BREAK or NOBREAK	NOBREAK
MODESW	Specifies whether or not the TERMINAL command or the STBREAK macro can be used to change the mode of operation specified with the MODE parameter.	YES or NO	NO
RCFBDUMP	Specifies the values of the return code (xx) and the feedback field (yy) for a RPL-type request. When an abnormal termination occurs that produces a return code and a feedback value that matches those specified in RCFBDUMP, a dump is taken to the extent specified in z. The specification z=0, specifies that only the local address space is to be dumped; z=1 specifies a full storage dump. The RCFBDUMP parameter can be used to request dumps for I/O errors from which TSO/VTAM can recover.	See return code and feedback values in *VTAM Programming*	None
CONFTXT	Specifies whether the buffer output is to be confidential. YES indicates that the buffer output is to be confidential and, as such, not traceable; the data in the VTAM buffers is overwritten with zeros immediately after it is sent to the terminal.	YES or NO	YES

Member Name: VATLSTxx (Volume Attribute List)

Use of the Member

VATLSTxx member(s) contain the volume attribute list(s) that predefine the mount and use attributes of direct access volumes. The *mount* attribute determines the conditions under which a volume can be demounted. The *use* attribute controls the type of requests for which a volume can be allocated. The volume attribute processor reads the volume attribute list(s) in order to set mount and use status bits in direct access UCBs[13].

The system programmer can predefine volume "mount" attributes as permanently resident or reserved, and can predefine volume "use" attributes as storage, public, or private. Therefore, critical direct access volumes can be controlled because the "mount" and "use" attributes determine the type of data sets that can be placed on a volume. During allocation, data sets on volumes marked permanently resident or reserved are selected first because they require no serialization, thus minimizing processing time.

You can ensure a faster initialization by specifying the volume attribute list(s) efficiently. Do not, for example, specify a list at a given IPL that contains entries for volumes that will not be mounted. Un-mounted volumes require operator intervention with resultant delay.

There are two ways to define the use and mount attributes for DASD volumes. You can define them in individual entries in the VATLSTxx member, one volume to each entry, or you can use one entry to define a group of volumes. In this second way, you specify generic volume serial numbers and device types for groups of volumes.

Specifying generic values makes it easier for you to maintain your VATLSTxx members. Give one generic name to a group of volumes mounted on devices that have the same device type and the same mount and use attributes. Specify an asterisk ("*") for the device type when groups of volumes have the same volume serial numbers, mount attributes, and use attributes, regardless of the devices the volumes are mounted on. See "Specifying a Generic Volume Serial Number" on page 3-243 and "Specifying a Generic Device Type" on page 3-244 for more information. Note that you cannot give generic volume serial numbers to MSS volumes.

For JES3, DEVICE and SETRES initialization statements, rather than the VATLST, can be used to specify permanently-resident volumes. In a JES3 complex, you must assign the same mount attribute to a volume mounted on every system in the complex. For example, if you use the mount attribute of "permanently resident" for the system residence volume on one system, you must use the same use attribute on all the other systems. Also, if a mount attribute of "reserved" is assigned to a volume that resides on a JES3-managed MSS virtual unit, that volume must be listed on a SETRES statement in the JES3 initialization deck. For more information, see *SPL: JES3 Initialization and Tuning*.

If VATLST members are used for volumes resident on devices that are shared among multiple systems, the mount attributes for a specific volume must be the same on all systems.

---

[13] Mount attributes are set in the UCBSTAT field, status byte A (offset 03) in the UCB. Use attributes are set in the UCBSTAB field, status byte B (offset 34 decimal, 22 hex) in the UCB.

Use of 3344 and 3350 Emulated 3330-1 and 3330-11 Devices

The category of devices consisting of the 3344 emulated 3340 and the 3350 emulated 3330-1 and 3330-11 must be made permanently resident via the VATLST facility because of a conflict in mount states. Because volumes on these emulated devices cannot be demounted, as can their real counterparts, the mount and demount messages that the operating system ordinarily issues for the real devices are erroneous. To prevent these erroneous messages, you must mark the emulated devices as permanently resident by so noting them in their respective VATLST entries.

Also, make sure that all emulated devices necessary for a given IPL are ready and available (not held in reserve by another CPU) at IPL time. Specifically, if another CPU has an emulated device reserved at IPL time, the operator must reply "WAIT" to the message "IEA120A DEVICE ddd SHARED. REPLY 'CONT' or 'WAIT'."

Definitions of the Mount and Use Attributes

There are three mount attributes: permanently resident, reserved, and removable. The permanently resident or reserved attributes may be specified in a volume attribute list. The removable attribute automatically applies to any volume that VATLSTxx does not designate or default as permanently resident or reserved.

A *permanently resident* volume is either one that can't be physically demounted (that is, a drum, 3344, or 3350) or one that can't be demounted until its device is varied offline. Only direct access volumes can be made permanently resident.

The following volumes are *always* marked permanently resident by NIP. You should therefore specify only the use attribute of these volumes in a volume attribute list:

- Volumes that can't be physically demounted (such as a 3350 volume).
- The system residence volume. This volume includes the SYS1.SVCLIB and SYS1.NUCLEUS data sets.
- Volumes that contain these system data sets: SYS1.LINKLIB and data sets concatenated to it, SYS1.DUMPnn, SYS1.STGINDEX, page data sets, and swap data sets.

Note: Although it is impossible to demount the devices physically, 3344 emulated 3340 devices and 3350 emulated 3330-1 and 3330-11 devices are *not* automatically marked permanently resident. You must instead mark their respective VATLST entries yourself.

A *reserved* volume remains mounted until the operator issues an UNLOAD or a VARY OFFLINE command. A volume is marked reserved when it is so designated in a volume attribute list, or when the operator issues a MOUNT command for the volume.

A *removable* volume can be demounted after its last use in a job, or when the device on which it is mounted is needed for another volume. Any volume not designated as either permanently resident or reserved is considered removable. The operator can change a removable volume to a reserved volume by issuing the MOUNT command for the volume.

The *use* attribute controls the type of request for which a volume can be assigned:

- a specific volume request
- a temporary, non-private non-specific volume request
- a non-temporary, non-private, non-specific volume request.

Three *use* attributes are used for allocating these types of volume requests, as follows:

- A *private* volume is allocated only to a specific volume request. For further information on this attribute, see *JCL*.
- A *public* volume is allocated to a temporary, non-specific volume request (or possibly to a specific volume request). Thus, a scratch data set would be placed on a public volume.
- A *storage* volume is allocated primarily to a non-temporary, non-specific volume request. (A storage volume can also be allocated to a specific volume request or a temporary non-specific volume request.)

Notes:

1. A storage volume is required by the SAVE subcommand of EDIT for a newly created data set. If a *storage* volume is not available, the SAVE subcommand cannot save the data set.
2. Because RESERVE and RELEASE commands are not supported by mass storage control, specify the volume mount and use attribute for the four mass storage control table volumes as RESERVED and PRIVATE.

For additional information on the mount and use attributes, see *SPL: System Modifications*.

Processing the VATLSTxx Member(s)

Volume attribute processing reads the VATLSTxx member(s) that were specified in the VAL parameter in IEASYSxx. If an invalid VATLST entry is detected, an informational message (IEA855I) is issued, and processing continues with the remaining entries.

If an I/O error occurs during the reading, the operator can choose to display an informational message (IEA850I) that lists all the volumes, device types, and attributes that will be processed. A second message (IEA853A) allows the operator to choose one of several recovery options:

- Continue processing any remaining lists.
- Stop the processing of remaining lists.
- Specify a new VATLSTxx member by replying r 0,xx, where xx is the two-character identifier for VATLSTxx.
- If necessary, re-IPL the system.

If an I/O error does not occur during the reading, the system lists the IPL and SYSTEM use attributes in message IEA168I.

Volume attribute processing compiles a list of all VATLSTxx entries. If MSS has not been initialized, MSS entries are ignored and not added to the list. If a particular volume serial number appears on more than one entry, the system uses the volume attributes specified in the last entry for that volume serial. If the volume

serial does not duplicate another entry, but is for an MSS volume (see the 'device type' parameter description), then the system uses the last entry with that particular MSS device number to set the volume attributes for the volume.

When volume attribute processing has set *mount* and *use* attributes for all mounted volumes specified in the VATLSTxx entries, it requests that the operator mount all un-mounted MSS volumes. When all MSS volumes have been mounted and their attributes set, the system issues mount messages (IEA851I and IEA851A) for entries that specify un-mounted volumes, unless mount message suppression was requested in the entries or you specified generic volume serial numbers or device types. Mount messages can be issued for un-mounted volumes, up to the maximum number of processed entries. As many as 4,096 unique VATLSTxx entries can be processed at one IPL. (Note that you cannot give volume serial numbers to MSS volumes.)

The operator may respond to the mount messages by replying with the device number of the requested device type (that is, 3330-1 or 2305-2). If the operator chooses to mount no volumes, the operator replies 'U' or ENTER.

If the device numbers that the operator enters are invalid (for example, the device number is invalid or a path to a device is not available), the operator can reenter new device numbers. Entering 'U' or ENTER indicates that no more volumes will be mounted.

When message IEA860A lists the devices that need volumes, the operator should mount the required volumes on the replied devices. When all devices have become ready (green lights on), the operator replies 'U' or ENTER to message IEA860A. Volume attribute processing then scans for mounted volumes.

If a volume that did not appear in the mount message is mounted on a unit specified by the operator, it is unloaded. The volume is also unloaded if the operator mounts the requested volume on a device type other than the one specified in the volume attribute list, or on an un-requested unit.

If all the required devices do not become ready, volume attribute processing issues message IEA893A, which lists the devices that are not ready. If the operator intends to ready these devices, the operator may do so before replying 'U' or ENTER to the message. If a volume cannot be mounted on a device for some reason (such as a hardware problem), the operator should reply NO to the message, after all other required devices have been processed. This response indicates to volume attribute processing that no more volumes will be mounted.

If the system does not find a volume that matches a generic volume serial number entry, it issues message IEA166I.

See *System Messages* for detailed information on volume attribute messages.

Parameter in IEASYSxx (or entered by the operator):

VAL= { **aa** | **(aa,bb,...)** }

Two alphameric characters (such as, A1 or 30) are appended to VATLST to specify the VATLSTxx member(s) of parmlib. If the parameter is not specified either in IEASYSxx or by the operator, the default member VATLST00 is used, if it exists. (VATLST00 is built by the installation, not through sysgen.) If the VAL parameter specifies multiple members, the members are processed in the order specified. If a

particular volume serial number appears on more than one entry, the volume attributes specified in the last entry for that volume serial will be accepted. If the volume serial does not duplicate another entry, but is for an MSS volume (see the 'device type' parameter description), then the last entry with that particular MSS device number will be used to set the volume attributes for the volume. If the VAL parameter has an invalid format, or if it specifies a member that doesn't exist in parmlib, the operator is prompted to respecify the member or to reply 'U' to cause the member to be ignored.

Creating a VATLSTxx Member

The VATLSTxx member contains an optional VATDEF statement, which specifies a default use attribute, followed by entries that define the mount and use attributes of DASD volumes.

Use the VATDEF statement in VATLSTxx to specify a default use attribute. If you do not specify VATDEF, the system assigns a default of *public* to those volumes that are not specifically assigned a use attribute in the VATLSTxx member or in a MOUNT command. VATDEF must be the first entry in the first VATLSTxx member you specify in the VAL= operand; the system ignores any later specifications of VATDEF that might appear in other VATLSTxx members.

The following is the syntax for the VATDEF statement. Unlike DASD volume entries in VATLSTxx members, the VATDEF statement is not column-dependent.

```
VATDEF
        IPLUSE ( { PUBLIC  } )
                 { PRIVATE }
                 { STORAGE }

        SYSUSE ( { PUBLIC  } )
                 { PRIVATE }
                 { STORAGE }
```

VATDEF
The default for use attributes for DASD volumes. If you specify VATDEF without any operands, the system uses the operands IPLUSE(PUBLIC) and SYSUSE(PUBLIC).

IPLUSE
The use attribute default that the system applies to permanently resident volumes that are brought online during IPL (that is, have no VATLST entry, or whose use attribute is not specified correctly in VATLSTxx).

SYSUSE
The use attribute default that the system applies to volumes that are varied online after IPL and have no entry in a VATLSTxx member.

PUBLIC
Sets the default use attribute to *public*. This operand is the default for VATDEF IPLUSE and VATDEF SYSUSE.

PRIVATE
Sets the default use attribute to *private*.

STORAGE
Sets the default use attribute to *storage*.

Example of Default Use Attributes: At an installation where the VATLSTxx members are defined as VAL=(00,05), VATLST00 member has the following entries:

```
VATDEF SYSUSE(PRIVATE)
30565A,0,2,3330    ,    TSO
```

The system ignores any VATDEF entry in VATLST05. It uses PRIVATE as the default use attribute for any volume varied online after IPL and PUBLIC for any volume brought online during IPL. The optional information TSO appears in installation printouts but is ignored by VATLST processing.

Syntax Rules

The following rules apply to the creation of a DASD volume entry in the VATLSTxx member:

- Each record consists of 80 columns, although columns 22 through 80 are ignored.
- The fields are column-dependent, as shown in "Internal Parameters."
- There are only two required fields: the volume serial number and the device types. All other fields have defaults.
- Use a comma to separate adjacent fields, except before the optional information field.
- Specify all characters in EBCDIC.

"Internal Parameters" includes a graphic description of all fields in the record and their lengths.

The following sections describe how to specify DASD volume entries.

Specifying a Generic Volume Serial Number

Generic volume serial numbers allow you to reduce the number of entries in the VATLSTxx members. The valid generic characters are % and *; you can use any combination of these characters in an entry. If a volume serial number does not contain any generic characters, it defines only the volume with the specified volume serial number. Rules for generic volume serial numbers follow, with examples of the generic numbers and the possible matching numbers.

Rules for Generic Volume Serial Numbers: The character "*" in the generic volume serial number indicates that any character in that position and all subsequent characters are a match:

Generic Volume Serial Number	Matching Volume Serial Numbers
TSO*	TSO01, TSO, and TSO123
TSO	XABTSO, ABTSO1, and TSO01
P*5	PROD05 and P5
*PROD	12PROD and PROD

The character "%" in the generic volume serial number indicates that any single character within the volume serial number is a match to that character position:

Generic Volume Serial Number	Matching Volume Serial Numbers
TSO%	TSO1
%TSO%	ATSO1
P%%5	P125
%PROD	1PROD

Both symbols can appear in the same number.

Generic Volume Serial Number	Matching Volume Serial Numbers
*TSO%	TSO1 and XXTSO2
P%5*	P05PRO

Specifying a Generic Device Type

The character "*" identifies a generic device type, based on which DASD are defined at your installation. Use the generic device type when you have the same use attribute and the same mount attributes regardless of the device you want the volumes mounted on.

If you specify a generic volume serial number, you can specify a generic device type. For example, if you specify the generic volume serial as TSO*, the matching volume serial numbers for this are TSO01, TSO, and TSO123, as above. TSO01 is on a 3330, TSO is on a 3350, and TSO123 is on a 3380. If you want to specify the same mount and use attributes for these volumes, regardless of the device types, you can specify a generic device type and then the mount and use attributes.

Example of Setting the Generic Device Type: The following VATLSTxx entry:

```
TSO*  ,0,2
```

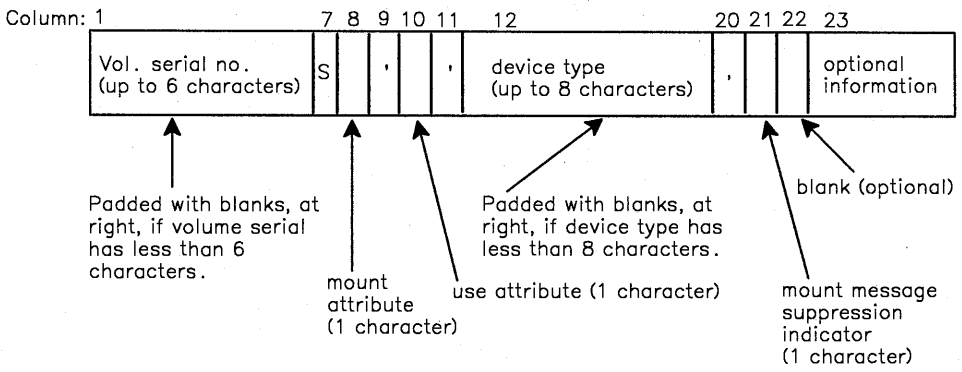
Is equivalent to the following three entries:

```
TSO01 ,0,2,3330
TSO   ,0,2,3350
TSO123,0,2,3380
```

These examples assume that only 3330, 3350, and 3380 DASD are defined.

Internal Parameters

The figure below shows the entries in VATLSTxx members that define the mount and use attributes of direct access volumes. The fields are column-dependent; they are described in the section that follows.



Examples of VATLSTxx entries: The following two examples show the entries in VATLSTxx member that define the mount and use attributes.

```
Columns:    1       8 10 12       21 23
            |       | |  |        |  |
            V       V V  V        V  V
30565A,0,2,3330bbbb,bb PAGING VOLUME
*TSO*,1,1,bbbbbbbb,bbTSO
```

In the first entry in the example, a volume whose serial number is 30565A is to be mounted on a 3330 and marked permanently resident. The volume's *use* attribute is to be *private*. A mount message is to be issued if the volume is demounted. The optional information, PAGING VOLUME, appears in installation printouts but does not affect VATLST processing.

In the second entry in the example, volumes with serial numbers that include the character string "TSO" are to be mounted on any DASD and reserved. The volumes' use attribute is to be *public*. Because you specified a generic volume serial number, the system does not issue a mount message for un-mounted volumes. The optional information, TSO, appears in installation printouts but does not affect VATLST processing.

Parameter	Column	Meaning and Use	Value Range	Default Value
volume serial	1-6	Specifies either the direct access volume whose *mount* and *use* attributes are to be set or a generic volume serial number. The volume serial number must begin at the first character position in the record.	1 to 6 alphameric characters plus special characters % and *, left justified in the field and padded with blanks at right to occupy six columns.	None
specific vol. ser. identifier	7	"S" specifies that the entry is a specific volume serial number, not a generic one. Code the "S" when your volume serial number has an asterisk in it and you do not want the system to process it as a generic entry.	S	None
mount attribute	8	Specifies whether the volume is to be permanently resident or reserved. Default is assigned if any character other than 1 is specified. **0** specifies permanently resident. **1** specifies reserved.	0 or 1	0
use attribute	10	Specifies whether the volume is to be defined as storage, public, or private. The default is assigned if any character other than 0, 1, or 2 is specified. **0** specifies storage. **1** specifies public. **2** specifies private.	0-2	1, unless the VATDEF operand specifies a different default.
device type	12-19	Specifies the device type, such as 3380. Up to eight characters may be specified, but the first character must start at column 12. Only supported device types that were defined through MVSCP are acceptable. This parameter indicates the basic device but does not denote special features, such as track overflow. An asterisk "*" in column 12 indicates a generic device type. **Note:** For 3330V volumes, do not specify "3330V" as you would specify another device. Instead, specify "Vxxx" where xxx is the device number of the 3330V device where the volume is to be mounted.	Up to 8 characters, left justified within the field, and padded with blanks at the right to occupy eight columns.	None

Parameter	Column	Meaning and Use	Value Range	Default Value
mount message suppression	21	Specifies whether mount messages should be issued for the volume if it is not already mounted. This parameter is ignored for MSS volumes and volumes that are defined with generic VATLST entries. **N** specifies that mount messages should be suppressed. **x** where x, which is a blank or any character except N, specifies that mount messages should be issued.	N, blank, or any character.	Blank, indicating that mount messages should be issued.
optional information	23-80	Contains any desired descriptive information. The information is not used by the system but does not appear in installation printouts.	Not applicable.	None

IBM-Supplied Defaults

No default member is supplied by IBM. The installation can, however, create its own default member, named VATLST00.

Part 4. Auxiliary Storage Management Initialization

This chapter gives guidelines for the effective use of page and swap data sets. Additional information on this subject appears under PAGE, SWAP, DUPLEX, NONVIO, and PAGTOTL parameters of parmlib member IEASYSxx.

Page/Swap Operations

The paging and swapping controllers of the auxiliary storage manager (ASM) attempt to maximize I/O efficiency by incorporating a set of algorithms to distribute the I/O load as evenly as is practical. In addition, every effort is made to keep the system operable in situations where a shortage of a specific type of page (swap) space exists. The following topics discuss some of the algorithms ASM uses to achieve these ends.

Paging Operations and Algorithms

To page efficiently and expediently, ASM divides the pages of the system into classes, namely PLPA, common, and local. Contention is reduced when these classes of pages are placed on different physical devices. Although the system requires only one local page data set, performance improvement can be obtained when local page data sets are distributed across on more than one device, even though some devices may be large enough to hold the entire amount of necessary page space. The PLPA and common page data sets are both required data sets, and there can be only one of each. Spillage back and forth between the PLPA and common page data sets is permissible, but, in the interest of performance, only spilling from PLPA to common should be tolerated.

Each local page data set is represented by a control block that is placed on one of three circular queues, according to the device on which the page data set resides. The three queues are:

1. The queue of local page data sets that are on cached auxiliary storage subsystems.
2. The queue of local page data sets that are on fixed-head devices.
3. The queue of local page data sets that are on movable-head devices.

The general intent of the ASM algorithms for page data set selection and channel program construction is to:

- *Prefer local page data sets on cached auxiliary storage subsystems:* ASM attempts to write first to the local page data sets on cached auxiliary storage subsystems, then to the local page data sets on fixed-head devices, and finally, to the local page data sets on movable-head devices.
- *Scan queues in a circular fashion:* ASM maintains three circular queues of control blocks representing the local page data sets. Each control block queue contains information about local page data sets for a particular type of device. When ASM writes a group of data, it selects a local page data set in a circular order, considering the availability of free space and device response time.
- *Write group requests to contiguous slots:* ASM selects contiguous space in local page data sets on moveable-head devices to receive group write requests, such as swap working set pages. For certain types of requests (such as a group write

request that results from a swap-out), ASM's slot allocation algorithm tries to select sequential (contiguous) slots within a cylinder. The reason for doing this is to shorten the I/O access time needed to read or write a group of requests. For other types of requests (such as an individual write request), or if there are no sequential slots, ASM selects any available slots.

- *Limit the apparent size of local page data sets to reduce seek time:* If possible, ASM concentrates group requests and individual requests within a subset of the space allocated to a local page data set.

  Note: ASM initiates I/O to the page data set using the suspend/resume function so that it does not have to repeatedly use the IOS STARTIO path. In using the suspend/resume function, ASM can monopolize a direct access storage device for its I/O and improve its performance for any of the requests going to page data sets.

If a duplex page data set is specified on a CLPA IPL, ASM writes secondary copies of the pages written to the PLPA and common page data sets on the duplex data set to provide greater system reliability. If a read request for either the PLPA or common page data set fails, ASM tries to read from the duplex page data set. If the read is successful, system operation remains uninterrupted. Whenever the duplex page data set becomes full, ASM suspends duplexing but can read any previously duplexed pages from that data set if the need arises.

If the user supplies no swap data sets, or if the swap data sets are temporarily filled, ASM writes the swap LSQA and private area working set pages to local page data sets.

Swap Operations and Algorithms

ASM sends LSQA and private area working set pages to swap data sets as long as the data sets are defined and contain free space. A swap data set consists of groups of 4096-byte slots called "swap sets." Each swap set consists of twelve contiguous slots. Swap data sets use only one seek per swap set. To ensure this seek efficiency, ASM prevents the swap set from crossing cylinder boundaries and uses the direct access device multi-track feature.

Each swap data set is represented by a control block that is placed on one of three circular queues, according to the device on which the swap data set resides. The three queues are: (1) the queue of swap data sets on cached auxiliary storage subsystems, (2) the queue of swap data sets that are on fixed-head devices, and (3) the queue of swap data sets that are on movable-head devices. During swap set selection, ASM prefers swap data sets on cached auxiliary storage subsystems, then swap data sets on fixed-head devices, and finally, swap data sets on movable-head devices. ASM scans each circular queue for a swap data set that has available swap sets. During this search, ASM gives preference to swap sets that:

- Have not yet been selected to fill the needs of the current request
- Are on a swap data set having no queued requests associated with it

If ASM finds no swap sets, it uses a local page data set.

ASM frees swap sets immediately upon swap-in; that is, swap pages are valid on the swap data set only for that period between swap-out and swap-in.

Page and Swap Data Set Sizes

Page and swap data set sizes can affect system performance. Note the following recommendations:

- *PLPA data set.* If the PLPA page data set is a high use data set, it should reside on a fixed-head device. However, if the common page data set has a higher use rate, it should be on the fixed head device in place of PLPA.

  The total combined size of the PLPA page data set and common page data set need not exceed the size of the PLPA (and extended PLPA) plus the amount specified for the CSA and the size of the MLPA. In defining the size of these data sets, a reasonable starting value might be four megabytes for PLPA and four megabytes for common, as spilling will occur if the PLPA data set becomes full. After the system is running, RMF reports can be used to determine the exact size requirements of these data sets.

- *Common page data set.* The common page data set should be large enough to contain all of the common area pages plus room for any expected PLPA spill. Although it is possible for the common page data set to spill to the PLPA page data set, this situation should not be allowed to occur because it may heavily impact performance. As noted for the PLPA data set, a reasonable starting size for the common page data set might be four megabytes. After the system is running, RMF reports can be used to determine the exact size requirements of this data set.

- *Duplex page data set.* The duplex page data set should be large enough to hold all the pages residing on both the PLPA and the common page data sets. After the system is running, RMF can be used to determine the exact size requirement of this data set.

- *Local page data sets.* The local page data sets must be large enough to hold all of the private area and those VIO pages which are not backed by expanded storage.

Space Calculation Examples

Figure 4-1 and Figure 4-2 show the values respectively for page and swap data sets. The examples following these figures show how to apply their tabular information to typical initialization considerations.

Note: After the system is running, you can use RMF reports to determine the sizes of page and swap data sets. RMF reports provide the minimum, maximum, and average number of slots in use for page and swap data sets. Thus, you can use the reports to adjust data set sizes, as needed.

Device Category	Device Type	Slots/Cyl	Cyl/Meg
cached auxiliary storage subsystem	3350	120	2.2
fixed head	2305-2	26	9.9
movable head	3340 3340-2 3350 3375 3380	24 24 130 96 150	10.7 10.7 2.0 2.7 1.8

Figure 4-1. Page Data Set Values

Device Category	Device Type	Slots/Cyl	Swap Sets/Cyl
cached auxiliary storage subsystem	3350	120	10
fixed head	2305-2	24 used	2
movable head	3340 3340-2 3350 3375 3380	24 24 120 96 150 144 used	2 2 10 8 12
Note: ASM uses space on a swap data set 12 slots at a time; all 12 slots must be in one cylinder. Therefore, the 3330 devices with 57 slots per cylinder available only use 48 (4 x 12) slots per cylinder for swap data sets.			

Figure 4-2. Swap Data Set Values

Example 1: Sizing the PLPA Page Data Set, Size of the PLPA Unknown

Define the PLPA page data set to hold four megabytes; if that amount of space is exceeded, the remainder can be placed on the common page data set until the PLPA value is determined exactly.

Therefore: From the tables, 8 cylinders on a 3380 are needed for the 4-megabyte PLPA page data set.

Example 2: Sizing the PLPA Page Data Set, Size of the PLPA Known

Assume the PLPA size is known to be 10 megabytes. Define the PLPA page data set to hold 10 megabytes plus 5%, or 10.5 megabytes. (The extra 5% allows for loss of space as a result of permanent I/O errors.)

Therefore: From the tables, 19 cylinders on a 3380 are needed for the 10.5-megabyte PLPA page data set.

Example 3: Sizing the Common Page Data Set

Assume that the exact size of the PLPA is unknown, but the PLPA page data set is defined to hold six megabytes. Start with a six-megabyte common page data set. After the system is running, you can use RMF reports to determine how much of the common page data set is being used and adjust the size of the data set accordingly.

Therefore: From the tables, 12 cylinders on a 3350 are needed to start with a 6-megabyte common page data set.

Example 4: Sizing the Duplex Page Data Set

The duplex page data set must be as large as the PLPA data set plus the common page data set.

Assume: The PLPA page data set and the common page data set each need 10 cylinders on a 3380.

Therefore: The duplex page data set will need 20 cylinders on a 3380.

Example 5: Sizing Local Page Data Sets

Assume that the master scheduler address space and JES address space can each use about eight megabytes of private area storage. Next, determine the number of address spaces that will be used for subsystem programs such as VTAM and the system component address spaces, and allow eight megabytes of private area storage for each. To determine the amount of space necessary for batch address spaces, multiply the maximum number of batch address spaces that will be allowed to be active at once by the average size of a private area (calculated by the installation or approximated at eight megabytes).

To determine the amount of space necessary for TSO, multiply the maximum number of TSO address spaces allowed on the system at once by the average size of a private area (calculated by the installation or approximated at eight megabytes).

Next, determine the amount of space required for any large swappable applications which run concurrently. Use the allocated region size for the calculation.

Finally, estimate the space requirements for VIO data sets. Approximate this requirement by multiplying the expected number of VIO data sets used by the entire system by the average size of a VIO data set for the installation. After the system is fully loaded, you can use RMF reports to evaluate the estimates.

For example purposes, assume that the total space necessary for local page data sets is:

```
    8   megabytes for the master scheduler address space
    8   megabytes for the PC/AUTH address space
    8   megabytes for the system trace address space
    8   megabytes for the global resource serialization address space
    8   megabytes for the allocation address space
    8   megabytes for the communications task address space
    8   megabytes for the dumping services address space
    8   megabytes for the system management facilities address space
    8   megabytes for the VTAM address space
    8   megabytes for the JES address space
    8   megabytes for the JES3AUX address space if JES3 is used
   10   megabytes for the batch address space (10 batches x 1 megabyte each)
   50   megabytes for TSO address spaces (50 TSO users x 1 megabyte each)
  102   megabytes for large swappable application
+  40   megabytes for VIO data sets (200 data sets x 0.2 megabyte each)
-------------------------------------------------------------------------
  290   megabytes total + 15 meg (approx. 5%) buffer = 305 megabytes
```

Therefore: From the tables, 549 cylinders on 3380 type devices are necessary.

Note: Even when the local page data sets will fit on one 3380, you should spread them across more than one device. See performance recommendation number 5. If large swappable jobs or large VIO users are started and there is insufficient allocation of space on local page data sets, a system wait X'03C' could result.

The installation should also consider the extent of the use of data-in-virtual when calculating paging data set requirements. Data-in virtual will be able to use sizable amounts of virtual storage which may put additional requirements on paging data sets.

Example 6: Sizing Swap Data Sets

Assume that six batch address spaces and 50 TSO address spaces constitute the maximum number of swappable address spaces to be active at any given time. At least four swap sets are generally necessary for each, and some may require five or more. Additionally, define a few extra swap sets in case of disk pack errors. A good starting value is therefore six swap sets times the maximum number of address spaces, or in this example:

56 x 6 = 336 swap sets

Therefore: From the table, 28 cylinders on 3380 type devices are needed.

Note: Although the swap data sets will fit on one 3380, you should spread them across more than one device. See performance recommendation number 5.

Performance Recommendations

The following recommendations might improve system performance through the careful use of paging data sets and devices:

1. Allocate only one paging data set per movable-head device (3380s, for example). Doing this reduces contention among more than one data set for the use of the device.

   Reason: If there is more than one paging data set on the same moveable-head device, a significant seek penalty will be incurred. Additionally, if the data sets are local page data sets, placing more than one on a device can cause the data sets to be selected less frequently to fulfill write requests.

   More than one paging data set on the same device also cause the loss of all gains in performance resulting from ASM using the suspend/resume function; all performance gained is lost because ASM will cause each I/O request for one page data set to interrupt the suspended I/O on the other data sets. This suspended I/O ends and must then be started again through the IOS STARTIO function.

   Comments: You might, however, place low-activity non-paging data sets on a device that holds a page data set and check for device contention by executing RMF to obtain direct access device activity reports during various time intervals. The RMF data on device activity count, percent busy, and average queue length should suggest whether device contention is a problem. RMF data for devices that contain only a page data set can be used as a comparison base.

2. Specify a data set to contain the primary copy of PLPA exclusively. You can do this by specifying the data set as the first dsname in the PAGE parameter in IEASYSxx or as the first dsname in the first DATASET macro containing the PAGEDSN parameter during sysgen. (The sysgen process places the dsname in PAGEDSN into IEASYS00.) Based on your storage contention, you can put the PLPA modules on the fastest available device because they are subject to high frequency use. If real storage management steals a PLPA module page that ASM needs later, ASM will subsequently have to read the page from the PLPA page data set. This read operation is faster if the PLPA page data set is on a fast device, for example a fixed-head device. Avoid placing other paging data sets on the same device.

   Note: The common page data set or local page data sets may be placed on the fastest device if either or both show a higher use rate than the PLPA page data set's use rate.

Reason: If the PLPA is a high use data set, it is desirable to speed the access to this data set and to avoid device contention.

3. Use swap data sets only if a configuration of all local page data sets proves to be inadequate.

   Reason: Because of ASM's selection of local page data sets and space management policies, a configuration of all local page data sets usually provides equal or better performance, and requires less tuning, than a configuration that contains swap data sets.

   Swap data sets might still prove useful when you:

   - Direct swap paging away from a particular device type. If a particular device handles non-swap paging more efficiently than swap paging, place swap data sets to other devices to force ASM to segregate the swap paging.
   - Prevent swap paging I/O from slowing down non-swap page-ins (or page faults). Sometimes a single page-in will be queued behind a swap I/O when swapping to local page data sets is occurring. If, in the rare case, page-ins are queued behind swap I/O with enough frequency to adversely impact page-in performance, use swap data sets to separate these two types of paging.

4. Place most paging data sets on moderate speed devices, such as 33xx's, to avoid overloading the faster devices. An exception to this general rule involves the swap data set or paging data sets for a large time sharing or teleprocessing installation for which response time is critical. Such uses may require the fastest device obtainable.

   Reason: The fastest devices should be used for the most critical applications. In general, this does not include local page data sets. However, to take advantage of the 33xx type of devices (reasonable speed plus size), ASM splits the paging load as evenly as possible across all defined local page data sets.

   To do this, ASM classifies the local page data sets according to the devices they are on. There are three classes of local page data sets: those that are on cached auxiliary storage subsystems, those that are on fixed-head devices, and those that are on movable-head devices. The control blocks for each of these classes are arranged in circular queues. While work is pending, ASM scans the queues and selects local page data sets, in order, from the cached auxiliary storage subsystem queue, then from the fixed-head device queue, and finally from the movable-head device queue.

   ASM considers the device response time when selecting a local page data set.

5. Over-specify space for all page data sets.

   Reason: Over-specifying space allows for the creation of additional address spaces before the deletion of current ones and permits some reasonable increase in the number of concurrent VIO data sets which may be backed by auxiliary storage. VIO data set growth might become a problem because there is no simple way to limit the total number of VIO data sets used by multiple jobs and TSO sessions. VIO data set paging can be controlled by restricting it to certain page data sets through the use of directed VIO; for example, specifying the NONVIO parameter for page data sets on the faster fixed head devices can direct VIO data set pages to other page data sets.

   VIO data set pages can be purged by a re-IPL specifying CVIO (or CLPA). CVIO indicates that the system is to ignore any VIO pages that exist on the page data sets and treat the page data sets initially as if there is no valid data on them (that is, there are no allocated slots). Thus, specifying CVIO prevents the warm

start of jobs that use VIO data sets because the VIO pages have been purged. (For additional space considerations, see the guideline for estimating the total size of paging data sets later in this chapter.)

In all cases, ASM avoids scattering requests across large, over-specified, page data sets by concentrating its activity to a subset of the space allocated.

6. Use more than one local page data set, each on a unique device, even if the total required space is containable on one device. Also use more than one swap data set, each on a unique device, even if the total required space for LSQA and private area working set pages can be contained on one device.

   Reason: When ASM uses more than one data set, it can page concurrently on more than one device. This is especially important during peak loads.

7. Distribute ASM data sets among channel paths and control units.

   Reason: Although ASM attempts to use more than one data set concurrently, the request remains in the channel subsystem queues if the channel path or control unit is busy.

8. Dedicate channel paths and control units to paging devices.

   Reason: In heavy paging environments, ASM can use the path to the paging devices exclusively for page-ins and page-outs and avoid interference with other users, such as IMS.

9. Make a page data set the only data set on the device.

   Reason: Making a paging data set the only data set on the device enables ASM to effectively use the suspend/resume function. ASM can monopolize the device to its best performance advantage by controlling its own I/O processing of that data set. ASM doesn't have to perform the additional processing it would otherwise have to perform if I/O for any other data set, especially another page data set, were on the same device. If another data set must be placed on the device, select a low-use data set to minimize contention with the page data set.

10. Do not share volumes that contain page or swap data sets among multiple systems.

    Reason: While page and swap data sets may be defined on volumes that contain shared non-paging data sets, they cannot be shared between systems.

Estimating Total Size of Paging Data Sets

You can obtain a general estimate of the total size of all paging data sets by considering the following space factors.

1. The space needed for the common areas of virtual storage (PLPA and extended PLPA, MLPA, and CSA). Double this space estimate if duplexing is desired.
2. The space needed for areas of virtual storage that are not duplexed: private areas of concurrent address spaces, and concurrently existing VIO page data sets. (The system portion of concurrent address spaces needs to be calculated only once, because it represents the same system modules.)

Using Measurement Facilities

You can possibly simplify the space estimation for the private areas mentioned above by picking an arbitrary value. Set up this amount of paging space, and then run the system with some typical job loads. To determine the accuracy of your estimate, start RMF while the jobs are executing. The paging activity report of the measurement program gives data on the number of unused 4K slots, the number of VIO data set pages backed by auxiliary storage, address space pages, and the number of unavailable (defective) slots.

The RMF report also contains the average values based on a number of samples taken during the report interval. These average values are in a portion of the report entitled "Local Page Data Set Slot Counts." They are somewhat more representative of actual slot use because slot use is likely to vary during the report interval. The values from the paging activity report should enable you to adjust your original space estimate as necessary.

Adding More Paging Space

To add more paging space, you 1) must use the DEFINE PAGESPACE command of access method services to pre-format and catalog each new page data set, and 2) can use the PAGEADD operator command to add the page data set or specify the data set at the next IPL by means of the PAGE parameter. (See *Access Method Services Reference* for information about the DEFINE PAGESPACE command, and on the related commands - ALTER and DELETE - used for the handling of VSAM data sets. See *Operations: System Commands* for a description of the PAGEADD command. Also see the description of the PAGE parameter in parmlib member IEASYSxx in "Part 3: System Initialization" in this manual.)

Deleting, Replacing or Draining Page Data Sets or Swap Data Sets

You might need to remove a local page data sets or swap data set from the system use for any of the following reasons:

- The hardware is being reconfigured.
- The hardware is generating I/O errors.
- The configuration of the page or swap data set is being changed.
- System tuning requires the change.

The PAGEDEL command allows you to delete, replace or drain local page data sets and swap data sets without an IPL. See *System Commands* for the description of the PAGEDEL command.

ASM will reject a PAGEDEL command that will decrease the amount of auxiliary storage below an acceptable limit. ASM determines what is acceptable by examining SRM's auxiliary storage threshold constant, MCCASMTI. If it is determined that too much storage would be deleted by the PAGEDEL command, ASM will fail the the page delete request.

Questions and Answers

Customers have often asked the following questions about ASM:

Q: *Does ASM use I/O load balancing?*

A: Yes, ASM does its own I/O load balancing.

When selecting a local page data set to fulfill a write request, ASM attempts to avoid overloading page data sets. ASM also attempts to favor those devices or channel paths that are providing the best service.

Q: *How does the auxiliary storage shortage prevention algorithm in SRM prevent shortages?*

A: It does so by swapping out address spaces that are accumulating paging space at a rapid rate. Page space is not immediately freed, but another job or TSO session (still executing) will eventually complete and free page space. SRM also prevents the creation of new address spaces and informs the operator of the shortage so that he can optionally cancel a job.

Q: *Is running out of auxiliary storage (paging space) catastrophic?*

A: No, not necessarily; it might be possible to add more page data sets via the PAGEADD operator command, optionally specifying the NONVIO system parameter. It may be necessary to re-IPL to specify an additional pre-formatted and cataloged page data set. (See the description of the PAGE parameter of the IEASYSxx member in "Part 3: System Initialization.")

Q: *Can we dynamically allocate more paging space?*

A: Yes. Additional paging space may be added via the PAGEADD operator command if the PAGTOTL parameter allowed for expansion (see the description of the PAGTOTL parameter of the IEASYSxx member in "Part 3: System Initialization," and the PAGEADD command in *Operations: System Commands*).

Q: *Can we remove paging space from system use?*

A: Yes. Use the PAGEDEL command for local page data sets or swap data sets.

Q: *How does ASM select slots?*

A: How ASM selects slots when writing out pages to page data sets depends on whether the write request is an individual request or a group request. For an individual write request, such as a request to write stolen pages (those pages changed since they were last read from the page data set), ASM selects any available slots. For a group write request, such as a request that results from a swap-out or a VIO move-out of groups of pages to page data sets, ASM attempts to select available slots that are contiguous. ASM also attempts to avoid scattering requests across large page data sets.

Q: *What is a cached auxiliary storage subsystem?*

A: A cached auxiliary storage subsystem consists of 3350 DASD that are attached to a 3880 Storage Control Model 11 or Model 21. The cache, which is in the control unit, contains the most recent copies of the frequently referenced pages of the active page and swap data sets on the DASD.

Q: *What are some of the benefits of using a cached auxiliary storage subsystem?*

A: In comparison to other DASD arrangements, which provide a single level of auxiliary storage, the cached auxiliary storage subsystem (because of its cache) provides a double level of auxiliary storage that is managed by the subsystem itself. For an installation, this should mean less attention to managing and tuning auxiliary storage space. Additionally, the cached auxiliary storage subsystem facilitates increased utilization of those channels to which it attaches. As a result of increased channel utilization, an installation should be able to reduce the number of channels dedicated to paging activity.

Q: *When does the cache in a cached auxiliary storage subsystem get initialized, and does cache initialization affect system initialization?*

A: The cache gets initialized:

- On each cold start or quick start IPL if page or swap data sets to be used reside on a cached auxiliary storage subsystem.
- On a warm start IPL if ASM determines that the cache requires initialization.
- When a PAGEADD command adds a page or swap data set to the system and the data set resides on a cached auxiliary storage subsystem that has not yet been initialized.

Although the time needed to initialize a cache is minimal, if your installation uses many cached auxiliary storage subsystems, the time needed to initialize all of the caches might be noticeable. For example, when the system initializes many caches on a cold start IPL, system activity might not be apparent during cache initialization. However, if cache initialization fails, the system issues an appropriate message to inform the operator.

Q: *Which page data sets and swap data sets can be placed on a cached auxiliary storage subsystem?*

A: In general, any page data set or swap data set can be placed on a cached auxiliary storage subsystem. However, consider the following:

- If a duplex page data set is placed on the same cached auxiliary storage subsystem as either the PLPA page data set or the common page data set, the back-up capability of the duplex page data set would be lost if a malfunction occurs in the control unit or the cache within the control unit.
- If warm start IPLs are frequently done and journaled VIO data sets are placed on a cached auxiliary storage subsystem, the journaled VIO data sets would be lost if a malfunction occurs in the control unit or the cache within the control unit or if the control unit is either IPLed or powered down.

Q: *How does ASM select a local page data set for a page-out?*

A: ASM selects a local page data set for page-out from its three queues of available page data sets. ASM first considers the queue of local page data sets on cached auxiliary storage subsystems, then the queue of fixed-head data sets, and finally the queue of moveable-head data sets. ASM selects these data sets in a circular order within each queue, subject to the availability of free space and the device response time.

Q: *How does ASM select a swap data set for swap-out?*

A: ASM selects a swap data set for swap-out from a list of swap data sets defined in the SWAP system parameter of the IEASYSxx parmlib member the installation uses during system initialization. In its selection process, ASM prefers swap data sets on cached auxiliary storage subsystems, then swap data sets on fixed-head devices, and finally, swap data sets on movable-head devices. ASM selects swap data sets on devices that have the shortest request queue and ignores swap data sets that have no available swap sets.

For each set of pages it wants to swap out, ASM searches the list of available swap data sets and uses the following criteria, choosing the first criteria as the most favorable and the last criteria as the least favorable, to select:

1. The swap data set that is not busy, was last used for a swap-in, and contains an available swap set. (When a swap-in has just been done from a swap data set on a movable-head device, it is possible that the arm will be positioned over the available swap set.)
2. The swap data set that is not busy and contains an available swap set.
3. The swap data set that is busy, was last used for a swap-in, and contains an available swap set.
4. The swap data set that is busy and contains an available swap set.
5. Any swap data set that contains an available swap set.

Note: If no swap data sets are defined or all swap space on the available swap data sets is in use, ASM writes the swap-out pages to the local page data sets.

Q: *How will extended storage configured on a 3090 model 200 processor complex affect my paging configuration?*

A: Extended storage helps reduce the paging and swapping load to auxiliary storage devices. Therefore, you might be able to reduce the paging configuration below what was previously required when no extended storage was configured.

With extended storage in use, some swap paging to auxiliary storage devices will be written to local page data sets, rather than to swap data sets. In particular, most non-LSQA, non-fixed, working set pages of migrated swap groups will be written to local page data sets. Therefore, you might need to adjust the relative proportion of swap data sets to local page data sets. A configuration of only local page data sets, in this case, might be the best choice.

Q: *Will data-in-virtual users increase the need for paging data sets?*

A: Data-in-virtual does provide applications with functions that would allow for extensive usage of virtual storage. Depending on the extent of the usage of data-in-virtual, paging data set requirements may increase.

Part 5. The System Resources Manager

Introduction

To a large degree, an installation's control over the functioning of the system is exercised through the system resources manager (SRM). The following sections describe the types of control available through SRM, the functions used to implement these controls, the concepts inherent in the use of SRM parameters, and the parameters themselves.

Guidelines are presented for defining these parameters and several complete sample specifications are described. Finally, a list of potential problems along with guidelines for evaluation and adjustment of parameters are provided.

System Tuning and SRM

The task of tuning a system is an iterative and continuous process. The controls offered by SRM are only one aspect of this process. Initial tuning consists of selecting appropriate sysgen parameters and parameters for various system components and subsystems. Once the system is operational and criteria have been established for the selection of jobs for execution via job classes and priorities, SRM will control the distribution of available resources according to the parameters specified by the installation.

SRM, however, can only deal with available resources. If these are inadequate to meet the needs of the installation, even optimal distribution may not be the answer — other areas of the system should be examined to determine the possibility of increasing available resources.

When requirements for the system increase and it becomes necessary to shift priorities or acquire additional resources, such as a larger processor, more storage, or more terminals, the SRM parameters might have to be adjusted to reflect changed conditions.

Section 1: Description of the System Resources Manager (SRM)

SRM is a component of the system control program. It determines which address spaces, of all active address spaces, should be given access to system resources and the rate at which each address space is allowed to consume these resources.

Before an installation turns to SRM, it should be aware of the response time and throughput requirements for the various types of work that will be performed on its system. Questions similar to the following should be considered:

- How important is turnaround time for batch work, and are there distinct types of batch work with differing turnaround requirements?
- Should subsystems such as IMS and CICS be controlled at all, or should they receive as much service as they need? That is, should they be allowed unlimited access to resources without regard to the impact this would have on other types of work?
- What is acceptable TSO response time for various types of commands?
- What is acceptable response time for compiles, sorts, or other batch-like work executed from a terminal?

(Guidelines for defining installation requirements are discussed in Section 3).

Once these questions have been answered and, whenever possible, quantified, and the installation is reasonably confident that its requirements do not exceed the physical capacity of its hardware, it should then turn to SRM to specify the desired degree of control.

Objectives

SRM bases its decision on two fundamental objectives:

1. To distribute system resources among individual address spaces in accordance with the installation's response, turnaround, and work priority requirements.
2. To achieve optimal use of system resources as seen from the viewpoint of system throughput.

An installation specifies its requirements for the first objective in a member of parmlib called the installation performance specification (IPS). Through the IPS, the installation divides its types of work into distinct groups, called *domains*, assigns relative importance to each domain, and specifies the desired control characteristics for each address space within these domains. The meaning and use of domains is discussed in detail in succeeding sections.

The installation control specification, another member of parmlib, assigns a set of IPS performance characteristics to each address space. The installation control specification can also be used to tailor the RMF reports for TSO, batch, and started tasks. These reports are useful for the continuous evaluation of system performance. RMF can also report on the transactions of subsystems that invoke the SRM reporting interface.

Another member of parmlib, the OPT member, contains parameters used to balance the use of the system's resources. Through a combination of IPS, OPT, and installation control specification parameters, an installation can exercise a degree of control over system throughput characteristics (objective number 2). That is, the

installation can specify whether, and under what circumstances, throughput considerations are more important than response and turnaround requirements when the need arises to make trade-offs between objective number 1 and objective number 2.

SRM attempts to ensure optimal use of system resources by periodically monitoring and balancing resource utilization. If resources are under-utilized, SRM will attempt to increase the system load. If resources are over-utilized, SRM will attempt to reduce the system load.

Types of Control

SRM offers four distinct types of control to an installation:

- Domain control
- Workload control
- Resource access control
- Throughput control

The installation control specification, IPS, and OPT contain the parameters used to modify these controls.

The remainder of this section describes these types of control and the functions used by the SRM to implement them.

Domain Control

Domains allow an installation to divide its types of work into distinct groups and thereby exercise individually tailored control over different types of work, such as batch work, IMS message processors, short TSO commands and long-running TSO commands.

Domains, therefore, are simply the means by which related types of work are grouped together, such that all work (address spaces) assigned to one domain has some common set of characteristics that differentiates it from the work assigned to other domains. What constitutes such a set of characteristics is entirely dependent on an installation's requirements. Work could be differentiated on the basis of execution characteristics such as short versus long-running and batch versus foreground, or according to different use characteristics such as IMS message processors and student or test programs. On the other hand, an installation may choose to use different user requirements as the basis for differentiating and divide its user population into domains, regardless of the execution characteristics of individual jobs. A mixture of the above considerations is, of course, equally possible.

Domain control enables an installation to do the following:

- Guarantee access to system resources to at least a minimum number of address spaces for each type of work.
- Limit the number of address spaces, for each type of work, that are given access to system resources.
- Assign degrees of importance to different types of work.

An installation may use these capabilities for various purposes, such as:

- To exercise either complete control, partial control, or no control at all over a particular type of work (or group of users).
- To prevent one type of work from competing with another type of work for access to system resources.
- To prevent one type of work from dominating the system.

The creation of domains, competition between domains, and the association of address spaces with domains are discussed in Section 2.

Workload Control

Address spaces compete for access to system resources with all other address spaces in the same domain. The installation establishes the rules for this competition via IPS parameters that allow specification of desired performance characteristics for each address space. These specifications are used by the workload management functions of SRM in apportioning system resources to individual address spaces.

An example of the competition within a domain would be a batch domain with different performance/throughput requirements for various initiator classes, such as payroll jobs versus test programs.

Another example of workload control is the setting of the average response time of short TSO transactions to a specified number of seconds.

Resource Access Control

SRM maintains control over a range of dispatching priorities called the automatic priority group (APG). By placing jobs in the APG range, further control over their performance characteristics is gained since this enables the installation, via the IPS, to alter the dispatching priorities of address spaces as their execution characteristics change.

Dispatching priorities control the rate at which address spaces are allowed to consume resources after they have been given access to these resources. This form of competition takes place outside the sphere of domain control, that is, all address spaces compete with all other address spaces with regard to dispatching priorities.

Another type of control is I/O request processing. High priority work may have its I/O requests processed ahead of low priority work on the same device (I/O priority queueing), or requests can be processed in a FIFO (first-in-first-out) manner. This choice is made through a parameter in the IPS.

Throughput Control

SRM attempts to maximize system throughput. If contention for a system resource is very high, bottlenecks may result, impeding throughput. For instance, when contention for the CPU becomes excessive, SRM will seek out an address space that is a major contributor to the problem and recommend that it be denied access to resources. Through an OPT parameter, the installation has the ability to specify the importance of such a load-adjusting recommendation. The installation has the option of specifying, via the IPS, whether throughput considerations should play a role or not.

Another example of throughput control is SRM's enqueue processing. A user may request a serial resource that is held by another user. In this case, SRM will ensure that the holder of the resource is in real storage and remains there for a fixed

execution time interval in the hope that the serial resource is released before the user is swapped out. The installation has the ability to specify this time interval through an OPT parameter.

Functions

This section discusses the functions used by SRM to implement the controls described in the earlier section.

The functions are as follows:

1. Swapping
2. Resource access control
3. Resource use functions
4. Enqueue delay minimization
5. I/O priority queueing
6. Device allocation
7. Prevention of storage shortages
8. Pageable frame stealing

Swapping

Swapping is the primary function used by SRM to exercise control over distribution of resources and system throughput. Using information specified by the installation via IPS and OPT parameters, and system status information that is periodically monitored, SRM determines which address spaces should have access to system resources.

In addition to the swapping controls described in the following text, SRM also provides an optional swap-in delay to limit the response time of TSO transactions.

There are seven reasons for swapping. The first three described below are used for control of domains and the competition for resources between individual address spaces within a domain, while the last four provide control over system-wide performance aspects and help increase the throughput.

1. *Unilateral swap in.* If the number of a domain's address spaces that are in real storage is less than the number the installation specified, or less than the number SRM considers optimal for the domain, SRM will swap in additional address spaces for that domain, if possible.
2. *Unilateral swap out.* If the number of a domain's address spaces that are in real storage is greater than the number the installation specified, or greater than the number SRM considers optimal for the domain, SRM will swap out address spaces from that domain.
3. *Exchange swap.* All address spaces of a domain compete with one another for system resources according to the installation's specifications in the IPS and OPT. When an address space in real storage has exceeded its allotted portion of resources, relative to an address space of the same domain waiting to be swapped in, SRM performs an exchange swap. That is, the address space in real storage is swapped out and the other address space is swapped in. This competition between address spaces is described in detail in Section 2.
4. *Swaps due to storage shortages.* Two types of shortages cause swaps: auxiliary storage shortages and pageable frame shortages. If the number of available auxiliary storage slots is low, SRM will swap out the address space that is acquiring auxiliary storage at the fastest rate. For a shortage of pageable

frames, if the number of fixed frames is very high, SRM will swap out the address space that acquired the greatest number of fixed frames. This process continues until the number of available slots rises above a fixed target, or until the number of fixed frames falls below a fixed target.

5. *Swaps due to wait states.* In certain cases, such as a batch job going into a long wait state (LONG option specified on the WAIT SVC, an STIMER wait specification of greater than or equal to 0.5 seconds, an ENQ for a resource held by a swapped out user), the address space will itself signal SRM to be swapped out in order to release storage for the use of other address spaces. Another example would be a time sharing user's address space that is waiting for input from the terminal after a transaction has completed processing. SRM also detects address spaces in a wait state. That is, address spaces in real storage that are not executable for a fixed interval will be swapped out. All address spaces swapped out because of wait states are candidates for logical swapping. (See "Demand Swapping" later in this section.)
6. *Request Swap:* The system may request that an address space be swapped out. For example, the CONFIG STOR, OFFLINE command requests the swap out of address spaces that occupy frames in the storage unit to be taken offline.
7. *Transition Swap:* A transition swap occurs when the status of an address space changes from swappable to nonswappable as a result of the TRANSWAP sysevent. For example, the system performs a transition swap out before a nonswappable program or V=R step gets control. This special swap prevents the job step from improperly using reconfigurable storage.

Resource Access Control

Dispatching of work is done on an address space priority basis. That is, those ready address spaces with the highest priority are dispatched first. When an address space is added to the dispatching queue, it is positioned according to its priority; if other address spaces with the same priority are already on the queue, it is added to the bottom of the group with that priority.

A job step or procedure is assigned a dispatching priority according to the value of the DPRTY parameter in the job's JCL specifications (see *JCL*). Installations might prefer to control the dispatching of work to enhance system throughput by overriding the priority specified in the JCL. SRM provides this ability with the Automatic Priority Group (APG). The APG is defined in the IPS.

The APG is that portion of the dispatching priority range that an installation intends to control dynamically. The total range of priorities is from 0 to 255. This range is divided into 16 sets of 16 priorities each. The APG can include any or all of these sets. If the JCL specifies a dispatching priority that falls within the installation-defined APG range, or a dispatching priority was not specified, the priority is obtained from the IPS.

Note: The master scheduler address space, the communications task address space, and the global resource serialization address space execute at the highest priority regardless of the APG range.

Within each of the 16 sets of priorities in the APG, the 16 priorities are grouped according to mean-time-to wait and fixed algorithms. Fixed has a higher priority than mean-time-to-wait. A description of the two algorithms follows.

Mean-time-to-wait

This algorithm can be used to increase system throughput by increasing CPU and I/O overlap. SRM periodically monitors each address space's CPU utilization by measuring the amount of CPU execution time between waits. Those address spaces that are considered to be CPU-bound are assigned lower dispatching priorities, within this group, than those which are considered to be I/O-bound. This permits the CPU-bound jobs to productively use the time spent waiting for I/O processing to complete

The SRM constant (CCCSIGUR) used to define heavy CPU users for the CPU load balancing function is also used in dispatching control to determine the range of utilization values that are associated with each of the ten levels of priority within the mean-time-to-wait group. This value can be changed by means of a parameter in the IEAOPTxx parmlib member.

Fixed

This algorithm simply assigns a priority to the address space based on what is coded in the IPS. SRM does not adjust this priority in any way.

Time Slicing

In combination with either algorithm, an address space's access to the system resources may also be controlled by time slicing.

Time slicing enables the installation to differentiate users (subsystems) into groups based on dispatching priority requirements. These groups are associated, via the IPS, with two priorities: a base priority and a time slice priority (such a group is referred to as a time slice group). SRM allots time slices to these groups on the basis of a pattern specified in the IPS. For the duration of a time slice, each address space in the respective time slice group is raised from its base priority to its time slice priority. When the time slice is completed, each address space is returned to its base dispatching priority.

The installation can control the impact of a time slice group over other units of work in the system. For instance, if an interactive subsystem at a high dispatching priority is interfering with batch throughput, the installation could use time slicing to periodically place the subsystem at a base dispatching priority below batch. The installation can specify the frequency and duration for the time slice as well as the percentage of time that the subsystem will receive its base and time slice dispatching priority. This enables the installation to maintain a reasonable response time for the subsystem without degrading the batch job throughput.

Resource Use Functions

The resource use functions of SRM attempt to optimize the use of system resources on a system-wide basis, rather than on an individual address space basis. The functions are as follows:

- Multiprogramming level adjusting
- CPU load balancing
- I/O load balancing
- Storage load balancing
- Demand (logical) swapping
- Selective enablement for I/O

The installation can influence or eliminate the effect of the multiprogramming level adjusting function via the parameter values of the domain specification in the IPS

and the MPL adjusting constants in the OPT. The installation can influence or eliminate the effect of the CPU, I/O, and storage load balancing functions with the response/throughput bias (RTB) in the IPS and the CPU, I/O, and storage load balancer coefficients in the OPT. The RTB parameter in the IPS can select one or more of the load balancing functions for a set of address spaces. SRM automatically performs demand swapping when sufficient real storage is available. The installation can influence or eliminate logical swapping via threshold values set in the OPT. The installation can influence the selective enablement for I/O function by adjusting constants in the OPT.

Multiprogramming Level Adjusting: SRM monitors system-wide utilization of resources, such as the CPU and paging subsystem, and seeks to optimize it by alleviating imbalances, that is, over-utilization or underutilization. This is accomplished by periodically adjusting the number of address spaces that are allowed in real storage (multiprogramming level) for appropriate domains.

For instance, when the paging rate becomes too high, it indicates that the percentage of CPU time used for productive (problem program) work is lower than it should be. That is, the system is being over-utilized. To reduce this excessive contention for storage, SRM will select a domain and reduce the number of address spaces allowed in real storage for that domain.

When system contention factors indicate that the system is not being fully utilized, SRM will select a domain and increase the number of address spaces allowed in real storage for that domain, thereby increasing utilization of the system.

CPU Load Balancing: This function attempts to maintain a dispatchable job mix that neither under-utilizes nor over-utilizes the CPU. SRM periodically monitors the system-wide CPU load to determine whether such imbalances exist.

The CPU is over-utilized if the utilization exceeds the high threshold (CCCUTHIT). The CPU is under-utilized when its utilization is less than the low threshold (CCCUTLOT). If, during the period under consideration, the processor did not enter the wait state and some swapped-in dispatchable user was not dispatched, the utilization is defined to be 101 percent.

SRM also monitors the CPU use of individual address spaces. A threshold mean execution time before entering the wait state (constant CCCSIGUR) defines a heavy CPU user.

The CPU utilization thresholds and the significant user threshold (CCCSIGUR) can be changed by means of parameters in the IEAOPTxx parmlib member. Note that CCCSIGUR also controls the assignment of dispatching priorities to users in the mean-time-to-wait groups.

When SRM determines that a CPU imbalance exists, it searches for heavy CPU users and calculates recommendation values for swap out (to correct over-utilization) or swap in (to correct underutilization). These values are based on the extent to which the CPU load is out of balance.

When the swap-in or swap-out of an address space has a positive effect on the processor utilization, the CPU load balancer continues the swap recommendation for that address space for a specified interval (constant CCCMNSWP). This action prevents the excessive exchange swapping that might otherwise occur as soon as the resource imbalance is corrected.

I/O Load Balancing: This function attempts to optimize the use of the channel subsystem by maintaining a mix of jobs that does not under-utilize or over-utilize the system's logical paths. A logical path is the set of all physical channel paths to a tape device or a direct access storage device (DASD). Although the logical path concept applies to all devices, SRM monitors logical paths only to tape devices and DASD.

Devices that are accessed by the same physical channel paths (and no other channel paths) are considered to have the same logical path. For example, if device numbers 360, 361, and 380 are each accessed by physical channel paths 03 and 12 (and no other channel paths), the devices have the same logical path, which could be called logical path 03,12.

However, if device number 380 is also accessed by physical channel path 23, but device numbers 360 and 361 are not, device number 380 would have a different logical path from device numbers 360 and 361. That is, device numbers 360 and 361 would be on logical path 03,12 while device number 380 would be on logical path 03,12,23.

SRM periodically calculates logical path utilization and accumulates device connect times for each logical path to determine if a path is under-utilized or over-utilized. A logical path that is either under-utilized or over-utilized is considered to be out of balance. To bring a logical path into balance, the I/O load balancing function recommends the swap in or swap out of significant users of the out-of-balance logical path. Therefore, SRM also monitors significant users of I/O. A significant user of a logical path is a user whose I/O device connect time to devices on the logical path exceeds a specified percentage (ICCSIGUP) of the total connect time for all devices on the logical path.

There are three classes of logical paths. Each class corresponds to a device type. The three logical path classes are:

- Logical paths for tape devices.
- Logical paths for DASD devices that have dynamic reconnect capability. (Such devices can reconnect to any available channel path in the logical path used by the devices.)
- Logical paths for DASD devices that do not have dynamic reconnect capability. (Such devices must reconnect to the same channel path initially selected for the I/O operation.)

To measure logical path utilization, SRM periodically samples each physical channel path to obtain its utilization and uses the channel path utilizations to compute logical path utilizations. Logical path utilization depends on the logical path class and is calculated as follows:

- For logical paths to tape devices or to DASD devices that have dynamic reconnect capability, the logical path utilization is the product of the channel path utilizations (for the channel paths belonging to the logical path). For example, if a logical path consists of two channel paths, one with a utilization of 80 percent and the other with a utilization of 20 percent, the logical path utilization is 16 percent (.80 x .20).

  The logical path utilization reflects the probability of finding all channel paths (of the logical path) busy at initial selection or at the time of reconnection for DASD.

- For logical paths to DASD devices that do not have dynamic reconnect capability, the logical path utilization is the average of the product of the channel path utilizations (for the channel paths belonging to the logical path) and the average of the sum of the channel path utilizations. For example, if a logical path consists of two channel paths, one with a utilization of 80 percent and the other with a utilization of 20 percent:

$$\frac{(.80 \times .20) + (.80 + .20) / 2}{2} = \frac{.16 + .50}{2}$$

  the logical path utilization is 33 percent.

  This calculation takes into account the fact that devices without dynamic reconnect capability must reconnect on the same channel path used at initial selection. The first factor in the average reflects the probability of finding all channel paths (of the logical path) busy at initial selection. The second factor reflects the probability that the channel path used at initial selection will be busy when a reconnect on that channel path is attempted.

To determine if a logical path is out of balance (under-utilized or over-utilized), SRM uses thresholds for the particular logical path class. A logical path is considered under-utilized when the logical path utilization is less than the low threshold for the logical path class. If a logical path is under-utilized, a significant user of that logical path will be favored for a swap in. A logical path is considered over-utilized when the logical path utilization is greater than the high threshold for the logical path class. If a logical path is over-utilized, a significant user of that logical path will be favored for a swap out.

When the swap-in or swap-out of a user has a beneficial effect on the logical path utilization (the logical path is being brought into balance), the swap recommendation for the user is not recalculated for a specified interval (constant ICCMNSWP). This action prevents the excessive exchange swapping that might otherwise occur as soon as the logical path imbalance is corrected.

The installation can change the thresholds that define when a logical path is considered out of balance and the percentage that defines significant I/O users by adjusting the appropriate parameters in the IEAOPTxx parmlib member. The logical path utilization parameters are ICCLPB(TAPE), ICCLPB(DPSDASD), and ICCLPB(NDPSDASD). The significant I/O user parameter is ICCSIGUP.

Note: SRM performs I/O load balancing if all of the following are true: (1) the I/O coefficient is not zero (IOC keyword in IEAOPTxx), (2) at least one performance group period defined in the IEAIPSxx parmlib member specifies I/O load balancing, and (3) the channel measurement facility is operational.

Storage Load Balancing: This function attempts to maintain a job mix that neither under-utilizes nor over-utilizes real storage.

When storage load balancing is specified, SRM periodically checks for a real storage imbalance. A real storage imbalance is either a shortage or an excessive availability of real storage. The indicators of a real storage imbalance are the long-term average steal criteria and the average number of frames on the available frame queue.

SRM considers real storage over-utilized if the following two criteria are met:

- The long-term average steal criteria is less than the MCCSBSTL low threshold.
- The average available frame queue length is less than the MCCSBATL low threshold.

SRM considers real storage under-utilized if the long-term average percentage of storage that is fixed, within the first 16 megabytes, is less than the MCCSBFTH threshold and one of the following two criteria are met:

- The long-term average steal criteria is greater than the MCCSBSTL high threshold.
- The average available frame queue length is greater than the MCCSBATL high threshold.

When SRM detects a real storage imbalance, it searches for significant storage users for which storage load balancing is applicable, and calculates a recommendation to swap out (to correct a storage shortage) or to swap in (to correct an excessive real storage availability). The swap recommendations are based on the degree of storage imbalance and the extent to which the address space uses real storage. If an address space does not exceed the threshold value for significant storage use (MCCSBSIG), it is not subject to storage load balancing.

When the swap-in or swap-out of an address space has a positive effect on the real storage utilization, the storage load balancer continues the swap recommendation for that address for a specified interval (constant MCCSBMSW). This action prevents the excessive exchange swapping that might otherwise occur as soon as the resource imbalance is corrected.

Storage load balancing improves throughput by delaying significant users of real storage until storage is available. When storage is under-utilized, transactions that require large amounts of storage remain in storage longer than they would in the absence of storage load balancing.

Address spaces require varying amounts of real storage based on factors such as processor speed and program design. Thus, the thresholds used to determine significant users of real storage must also be variable. SRM periodically increases or decreases the threshold to keep the ratio of significant to insignificant users of real storage at a constant target ratio (MCCSBSGP). Only address spaces that are subject to storage load balancing are considered when calculating this ratio. No users below this threshold are ever considered significant users of real storage. This threshold is initialized to a value (MCCSBSIG) and then, as the actual ratio deviates from the target ratio, it is adjusted either up (MCCSBINP) or down (MCCSBDEP) by a percentage of the initial value with the restriction that it is never adjusted below the initial value. The values for MCCSBSIG, MCCSBSGP, MCCSBINP, and MCCSBDEP can be changed by means of parameters in the IEAOPTxx parmlib member.

Demand Swapping: In order to use real storage more effectively and reduce processor and channel subsystem overhead, the SRM demand (logical) swap function attempts to prevent the automatic physical swapping of a TSO address space after a "terminal wait" or any address space after a "long wait" or "detected wait." Unlike a physically-swapped address space, where the LSQA, fixed frames, and recently-referenced frames are placed on auxiliary storage, SRM keeps the frames that belong to a logically-swapped address space in real storage.

A TSO user's address space is eligible to be logically swapped out whenever the user's think time (the time interval between the system's readiness for new input and the terminal user's response) is less than the system threshold value. SRM adjusts this threshold value according to the demand for real storage. SRM uses the unreferenced interval count (UIC) and the available frame queue length to measure this demand for real storage. As the demand for real storage increases, SRM reduces the system threshold value; as the demand decreases, SRM increases the system threshold value.

SRM periodically evaluates the system think time threshold and based on the demand for storage either increases the threshold by 0.5 sec., decreases it by 1.0 sec., or leaves it unchanged. Because the demand for storage can change abruptly when the workload changes, SRM also tests the system high UIC to determine if an address space should be logically swapped. By testing the UIC, SRM avoids logically swapping address spaces when the contention for storage suddenly increases.

The system threshold value for think time fluctuates between low and high boundary values. The installation can change these boundary values in the IEAOPTxx parmlib member. The installation can also set threshold values for the UIC and available frame queue length; setting these threshold values affects how SRM measures the demand for real storage.

SRM logically swaps out other address spaces that are in a "long wait" or a "detected wait" when the system threshold value for think time exceeds 5 seconds and the current UIC is greater than the UIC threshold values. SRM also logically swaps out address spaces that have been selected to be swapped out to expanded storage if the real frames they own are not required to immediately replenish the supply of available real frames.

SRM's logical swapping function periodically checks all logically swapped out address spaces to determine how long they've been logically swapped out. SRM physically swaps out those address spaces that have been logically swapped out for a period greater than the system threshold value for think time plus an internally-defined grace period of two seconds.

Selective enablement for I/O: Selective enablement for I/O is a function that SRM uses to control the number of processors that are enabled for I/O interruptions. The intent of this function is to enable only the minimum number of processors needed to handle the I/O interruption activity without the system incurring excessive delays. That is, if one processor can process the I/O interruptions without excessive delays, then only one processor need be enabled for I/O interruptions.

At system initialization, one processor is enabled for I/O interruptions. To determine if a change should be made to the number of processors that are enabled, SRM periodically monitors I/O interruptions processed by the I/O interrupt handler with the test pending interrupt (TPI) instruction. SRM computes the percentage of the total number of I/O interruptions processed by the TPI instruction.

By comparing this value to threshold values, SRM determines if another processor should be enabled or if an enabled processor should be disabled for I/O interruptions. If the computed value exceeds the upper threshold, I/O interruptions are being delayed, and another processor (if available) will be enabled for I/O interruptions. If the value is less than the lower threshold (and more than one processor is enabled), a processor will be disabled for I/O interruptions. The

installation can change the threshold values using the CPENABLE parameter in the IEAOPTxx parmlib member.

Note: In addition to enabling a processor when I/O activity requires it, SRM also enables another processor for I/O interruptions if an enabled processor is taken offline or has a hardware failure.

Enqueue Delay Minimization

This function deals with the treatment of address spaces enqueued upon system resources that are in demand by other address spaces or resources for which a RESERVE has been issued and the device is shared. If an address space controlling an enqueued resource is swapped out and that resource is required by another address space, SRM will ensure that the holder of the resource is swapped in again as soon as possible.

Once in real storage, a swap out of the controlling address space would increase the duration of the enqueue bottleneck. Therefore, the controlling address space is given a period of CPU service during which it will not be swapped due to service considerations (discussed in Section 2). The length of this period is specified by the installation by means of a tuning parameter called the enqueue residence value (ERV), contained in parmlib member IEAOPTxx.

I/O Priority Queueing

I/O priority queueing is used to control deferred I/O requests. If this function is invoked, all deferred I/O requests, except paging and swapping, will be queued according to the I/O priorities associated with the requesting address spaces. Paging and swapping are always handled at the highest priority. An address space's I/O priority is by default the same as its dispatching priority. All address spaces in one mean-time-to-wait group fall into one I/O priority. In addition, address spaces that are time sliced have their I/O queued at their time slice priority. Changes to an address space's dispatching priority when the address space is time sliced up or down, do not affect the I/O priority.

An installation can assign an I/O priority that is higher or lower than the dispatching priority for selected groups of work. For example, if the installation is satisfied with the dispatching priority of an interactive application but would like the application's I/O requests to be processed before those of other address spaces executing at the same priority, the application could be given an I/O priority higher than its dispatching priority.

If I/O priority queueing is not invoked, all I/O requests are handled in a first-in/first-out (FIFO) manner.

Device Allocation

This function attempts to balance the utilization of all logical paths by selecting the proper device for allocation to an address space from a list of candidate devices. Device allocation selects candidates for permanent data sets on mountable devices (JCL specifies nonspecific VOLUME information or a specific volume and the volume is not mounted).

Device allocation supports the I/O load balancing function. Device allocation tries to select a device on a logical path with low utilization while the I/O load balancing function makes swap recommendations to keep logical path utilization within predetermined bounds.

The ability of SRM to control device allocation is limited by the decision an installation makes at system generation time and at initial program loading (IPL) time, as well as by the user's JCL parameters. SRM can only apply its selection rules to a set of devices that are equally acceptable for scheduler allocation. This set of devices does not necessarily include all the devices placed in an esoteric group during system generation. At that time, an esoteric group is defined by the UNITNAME macro instruction and entered in the eligible device table (EDT). System generation also partitions each esoteric group into subgroups if either of the following conditions occurs:

- The group includes devices that are common with another esoteric group.
- The group includes devices that have certain generic differences (such as single and dual density tape drives). System generation partitions only esoteric groups that consist of magnetic tape or direct access devices.

For example, assume that you specify the following at system generation time:

```
UNITNAME=TAPE,UNIT=((470,7),(478,8),(580,6))
UNITNAME=IMSLOG,UNIT=((580,6))
```

Because of the intersection with IMSLOG (580,6), the TAPE group is divided into two subgroups: (470,7) and (478,8) in one subgroup and (580,6) in the other.

Allocation allows SRM to select from only one subgroup at a time. After allocating all devices in the first subgroup, allocation selects devices from the next subgroup. Using the previous example, when a job requests UNIT=TAPE, allocation tells SRM to select a device from the first group (470-476 and 478-47F) regardless of the relative use of channel paths 4 and 5. After all of the devices in the first group have been allocated, allocation tells SRM to select devices from the second group (580-585). The sysgen output describes the subgroups that allocation creates within the esoteric groups. Examine this output to determine the limitations of SRM's device selection algorithm.

A summary of the most important factors influencing device allocation are as follows:

- At system generation time
  - The IODEVICE statement in MVSCP
  - The UNITNAME statement in MVSCP
- At IPL time
  - For direct access devices, the number and placement of online, mountable devices
  - For tape devices, the number of drives in the configuration that have compatible densities
- At job initiation time
  - The DD JCL statement, specifically the UNIT and VOL=SER parameters
  - The number of unallocated devices
  - The requirements of other DD statements
- In JES3 installations, I/O devices are managed by JES3 and/or MVS. See *SPL: JES3 Initialization and Tuning* for an explanation of how resource definition affects JES3 resource management.

Because device allocations tend to be made at the beginning of a job's execution, the future impact of that job's I/O activity on logical path utilization is not known when the allocations are made. Therefore, simply choosing the least utilized logical path would cause all or most allocations for a job to be made to the same logical path. This effect is known as 'clumping'. The device allocation function attempts to avoid this effect by assuming that each data set will have a fixed impact on logical path utilization. Thus, when making an allocation choice for a user, a constant (ICCELPUT) is added to the measured utilization of a logical path for each data set previously allocated to the job on that path.

Device Selection Rules

SRM applies the two following standard selection rules to direct access and tape device selection:

- For tape devices, eliminate all ready devices (devices with volumes mounted) from a list of eligible devices. If all the eligible devices are ready and this is a tape request, eliminate all devices with private, passed or RETAINed volumes.
- For tape and direct access devices, construct a new list of those eligible devices on the logical path(s) with the lowest utilization relative to the high utilization threshold for the logical path(s).

SRM then applies one of the following rules to the new list:

- For direct access devices, SRM selects the devices with the lowest delay time and, from these, selects one at random.

  If the selected device is already allocated to this user, SRM constructs a new list that includes only those devices not already allocated to this user. All of the selection rules are then re-applied to the new list. If all eligible devices have been allocated to this user, a device is selected at random from the original list.
- For tape devices, SRM selects the device with the next highest device number from the last device number picked by SRM. The method of selection may be changed using the tape device selection options. For more information, see "Tape Device Selection Options" under "OPT Concepts" in Section 2.

Prevention of Storage Shortages

SRM periodically monitors the availability of three types of storage and attempts to prevent shortages from becoming critical. The three types of storage are:

- Auxiliary storage
- Pageable frames
- SQA

Auxiliary storage: When more than a fixed percentage (constant MCCASMT1) of auxiliary storage slots have been allocated, SRM reduces demand for this resource by taking the following steps:

- LOGON, MOUNT and START commands are inhibited until the shortage is alleviated.
- Initiators are prevented from executing new jobs.
- The number of address spaces allowed in real storage for each domain is set to its minimum value (see "Domains" in Section 2).
- The operator is informed of the shortage.

- Choosing from a subset of swappable address spaces, SRM swaps out the address space(s) acquiring slots at the fastest rate. When SRM swaps-out an address space because of excessive slot usage, SRM informs the operator of the name of the job that is swapped out, permitting the operator to cancel the job.
- SRM prevents all unilateral swap-ins (except for domain zero). This action allows the operator to cancel jobs or add auxiliary paging space to alleviate the problem.

If the percentage of auxiliary slots allocated continues to increase (constant MCCASMT2), SRM informs the operator that a critical shortage exists.

When the shortage has been alleviated, the operator is informed and SRM halts its efforts to reduce the demand for auxiliary storage.

Pageable Frames: SRM attempts to ensure that enough pageable real storage is available to the system. SRM monitors the amount of pageable storage available, ensures that the currently available pageable storage is greater than a threshold and takes continuous preventive action from the time it detects a shortage of pageable storage until the shortage is relieved.

Regardless of the cause of a shortage of pageable storage, SRM:

- Inhibits LOGON, MOUNT, and START commands until the shortage is relieved
- Prevents initiators from executing new jobs
- Informs the operator that a shortage of pageable storage exists

Further SRM actions to relieve the shortage depend on the particular cause of the shortage.

Several system conditions can cause a shortage of pageable storage:

- Too many address spaces are already in storage.
- Too much page fixing is taking place. One or more address spaces are using substantial amounts of fixed storage.
- Too much paging I/O is causing an unusually high number of frames to be made non-pageable during the I/O operations.

Too many address spaces in storage does not usually, of itself, cause a shortage of pageable storage because SRM performs MPL adjustment and logical swap threshold adjustment, which generally keep an adequate amount of fixed storage available to back the address spaces. (Refer to the section on "OPT Concepts" for information on MPL load adjustment and logical swapping.)

A shortage of pageable storage resulting from too much page fixing only applies to storage below 16 megabytes. A shortage of pageable storage resulting from both too much fixed storage and too much paging I/O, however, applies to all of real storage. Because of this distinction between these two causes of a shortage of pageable storage, SRM uses two unique thresholds to detect these two types of shortages. The thresholds that SRM uses can be modified by parameters in the IEAOPTxx parmlib member.

If too much page fixing is the cause of the shortage of pageable storage, SRM:

1. Identifies the largest swappable user or users of fixed storage. If any of these users own more frames than three times the median fixed frame count, SRM begins to swap them out, starting with the user with the largest number of fixed pages. SRM continues to swap users out until it releases enough fixed storage to relieve the shortage.
2. Begins to physically swap out the logically-swapped out address spaces if swapping out the largest users of fixed storage does not relieve the shortage of pageable storage.
3. Decreases the MPLs for those domains that have the lowest contention index if physically swapping out the logically-swapped out address spaces does not relieve the shortage of pageable storage. This MPL adjustment allows the swap analysis function of SRM to swap out enough address spaces to relieve the shortage.

If too much fixed storage and too much paging I/O is the cause of the shortage of pageable storage, SRM swaps out the address space that is using the most real storage.

SRM takes the following additional actions if the shortage of pageable storage reaches a critical threshold:

1. Informs the operator that there is a critical shortage.
2. Repeats all the steps described earlier that are applicable to the cause of the shortage (whether it be paging or fixing).
3. Prevents any unilateral swap-in, except for domain zero.

When the shortage of pageable storage is relieved, SRM:

- Allows new address spaces to be created through the LOGON, MOUNT, and START commands.
- Notifies the operator that the shortage is relieved.
- Allows initiators to process new jobs.
- Allows unilateral swap-ins.

Note: Those address spaces that SRM swapped out to relieve the shortage of pageable storage are not swapped back in if their storage requirements would potentially cause another shortage to occur.

SQA: When the number of available SQA and CSA pages falls below a threshold, SRM:

- Inhibits LOGON, MOUNT, and START commands until the shortage is alleviated.
- Informs the operator that an SQA shortage exists.

If the number of available SQA and CSA pages continues to decrease, SRM informs the operator that a critical shortage of SQA space exists and, except for domain zero, SRM prevents all unilateral swap-ins.

When the shortage has been alleviated, the operator is informed and SRM halts its efforts to prevent acquisition of SQA space.

Note: For a discussion of possible operator response to the messages described above, see SRM section IRAxxxx messages, in the publication *System Messages*.

Pageable Frame Stealing

Pageable frame stealing is the process of taking an assigned frame away from an address space to make it available for other purposes, such as to satisfy a page fault or swap in an address space.

When the demand for pageable frames is excessive, SRM will steal those frames that have gone unreferenced for the longest time and return them to the system. The unreferenced interval count (UIC) of each frame indicates how long it has been since an address space referenced it. This count is updated periodically by the SRM. When pages must be stolen, SRM distributes the stealing over as many address spaces as possible. Each address space, along with the common service area (CSA) and the pageable-link pack area (PLPA) is examined. If there is still a shortage of frames after all address spaces have been examined, the process is repeated using a lower UIC steal threshold.

The storage isolation function can be used to modify the stealing process for selected users or the common area by either protecting frames from being stolen or favoring them for stealing.

Stealing takes place strictly on a demand basis, that is, there is no periodic stealing of long-unreferenced frames. When SRM selects an address space to swap in and there are insufficient unused frames available for the address space, stealing takes place; when there are enough unused frames available for the address space, SRM swaps in the address space without having to steal pages.

Section 2: Introduction to SRM Parameter Concepts

This section discusses the concepts inherent in the installation control specification, IPS, and OPT parameters that control SRM's distribution of system resources to individual address spaces. Parameter descriptions and syntax rules are provided in Section 5 and should be referred to when necessary.

The section entitled "Guidelines and Examples" discusses recommendations for the selection of specific IPS and OPT parameter values, but these will not be meaningful unless the ideas presented in this section are understood.

Although care has been taken to choose typical numbers for the examples in this section, these numbers are not intended as guidelines for actual cases.

The parameters are presented in the following order:

- IPS concepts:
  - Domains
  - Service definition coefficients
  - Service rates
  - Performance objectives and workload levels
  - Interval service values
  - Dispatching priorities
  - I/O priorities
  - Storage isolation
  - Response-throughput bias
  - TSO response time control
  - Domain importance
  - Performance periods
  - Performance groups
- Installation control specification concepts:
  - Performance group assignment for control
  - Performance group assignment for reporting
- OPT concepts:
  - Resource load balancing coefficients
  - Tape device selection options
  - Special options
  - Adjusting constants options

In addition to these parameters, the meaning of terms such as service, service units, time interval, and transaction is discussed in detail where necessary to support the discussion of related parameters.

This entire section should be studied as one continuous presentation because the ideas introduced under each topic are dependent on the concepts, terms, and examples presented under the preceding topic. The examples, for instance, are continuously expanded with each newly introduced parameter until, under the topic "Performance Group," a complete IPS has evolved. At that point, the official syntax is used for the first time, requiring the reader to refer to Section 5.

The discussion of installation control specification and OPT concepts is also dependent on preceding topics, that is, on IPS parameter concepts.

It is assumed that the reader is familiar with the basic mathematical concepts used to plot the relationship between dependent and independent variables on a graph.

IPS Concepts

The parameters discussed in this part are specified in the IEAIPSxx member of SYS1.PARMLIB.

Domains

Domains, as discussed under "Domain Control" in Section 1, are a means to differentiate one type of work (or user) from another and thereby make it possible to establish different, individually suited, control over different types of work. This control is derived from the ability to specify the number of address spaces that should be allowed in real storage at one time for a domain. This number is called the *multiprogramming level,* or MPL, of a domain. In the IPS, the installation specifies the desired range for each domain's multiprogramming level, that is, the minimum and the maximum MPL (minMPL and maxMPL, for brevity).

SRM periodically computes for each domain an optimal MPL called the *target MPL,* adjustable up or down in response to changing activity levels.

The target MPL is the basis for SRM domain control since it represents the actual number of a domain's address spaces that are allowed in real storage at any one time.

The range of the target MPL is subject to the installation-specified constraints (minMPL and maxMPL), that is, it cannot be less than minMPL or greater than maxMPL, even when the computed optimal value exceeds these bounds. The installation, therefore, has the ability to select the degree of control it wishes to exercise over each domain, ranging from complete control to no control at all. More control by the installation, of course, implies less control by SRM.

Complete installation control is achieved by setting minMPL equal to maxMPL, resulting in a "fixed" target MPL and therefore eliminating SRM's ability to make MPL adjustments in response to changing activity levels.

Partial installation control is achieved by choosing different values for minMPL and maxMPL, thereby allowing room for SRM adjustment of the target MPL.

No installation control at all, or full SRM control, is the result of setting minMPL and maxMPL to their extreme values: minMPL = 0 and maxMPL = 999 (see "Section 5: SRM Parameters" for value ranges).

Domain Examples: Some basic examples will now be discussed to illustrate the purpose and use of domains and the effects achieved by the selection of particular constraints.

Example 1: A domain could be created for work that, for reporting purposes, should not be mixed with other types of work. For instance, nonswappable address spaces, or address spaces associated with different subsystems, could be grouped into unique domains. Because the resource measurement facility (RMF) provides workload information on a domain basis, placing such work into unique domains provides distinction for analysis purposes. That is, it enables the installation to determine how much of the total system is being used by a particular nonswappable address space, or by a particular subsystem.

Also, because APG priorities can be assigned to nonswappable address spaces, the effects on performance resulting from changes to the dispatching priority of such an address space can be evaluated by examining the RMF workload report for the respective domain.

Example 2: A domain could be created for work that has particularly fast response time or turnaround requirements, such as short (interactive) TSO commands. Placing such work into a unique domain with minMPL equal to the maximum number of ready users will ensure that each address space is swapped in as soon as it becomes ready. That is, forcing the target MPL to be equal to the maximum number of ready address spaces guarantees all address spaces of that domain immediate access to real storage.

Example 3: A domain could be created for users that might dominate the system. The maxMPL value can be used to limit the number of address spaces of this type of work allowed in real storage at one time.

For example, TSO users executing sorts or compiles from the terminal can slow down the system considerably. Placing such users into a domain with minMPL = maxMPL = 1 would have the effect of serializing such work, since only one address space of this type would be allowed in real storage at one time.

Batch work could be controlled in the same manner. Placing all batch work into a unique domain with maxMPL = 2 will limit the number of batch jobs concurrently in real storage to two, even if, for example, eight initiators were started.

Example 4: The MPL adjustment function can raise the target MPL of a domain unless prevented from doing so by the domain's constraints. The raising of the MPL target is limited within the domain's constraints as follows: the sum of the maximum ready users in the domain plus the average number of ready users in the domain; divided by 2. The adjustment function is most effective when the arrival rate of work is somewhat consistent, that is, when the number of ready address spaces over a time interval does not vary sharply.

The arrival rate of batch work, for example, is fairly consistent, since it is approximately equal to the number of started initiators. TSO work, however, can exhibit drastic fluctuations in arrival rate, causing variations in response times if the MPL adjustment function is allowed to determine the target MPL.

Service

One of the basic functions of SRM is to monitor the dynamic performance characteristics of all address spaces under its control in order to ensure distribution of system resources as intended by the installation.

A fundamental aspect of these performance characteristics is the *rate* at which an address space is receiving *service* relative to other address spaces competing for resources within the same domain.

Before discussing the meaning of service, a brief explanation of terminology is needed to avoid confusion. We have associated performance characteristics with address spaces. It is, of course, not the address space that has performance characteristics but the transaction associated with the address space. A transaction is simply SRM's way to measure the service consumption within an address space, that is, it is a way to delineate a unit of work that is consuming service. For batch jobs, a transaction corresponds to a job or job step. For TSO work, a transaction normally corresponds to a command or terminal interaction. (See Note.)

Since only one transaction can be active in an address space at one time, and it is indeed the address space that is swapped, the use of the term "address space" is more convenient when the emphasis of the discussion, for example, is on swapping. For the remainder of this section, the two terms will be used interchangeably, whichever is more appropriate.

Note: A new transaction is defined for a batch job step if it is the first step of the job, or if the performance group number (PGN, discussed later) is different from the previous job step's PGN.

A new transaction is defined for an active TSO user whenever

1. Terminal input is entered and the line is not continued.
2. The 3270 field mark key separates commands on the same input line.
3. A command's output is detained while waiting for an output buffer.
4. The OPT specifies CNTCLIST = YES and a command is taken from the TSO internal stack (as with a command in a CLIST). (Only if TSO Extensions is installed).

A CLIST is one transaction, unless case 3 or 4 applies. Every command typed ahead on an unlocked keyboard results in a new transaction.

The amount of service consumed by an address space is computed by the formula:

service = (CPU x CPU Service Units)
+ (IOC x I/O Service Units)
+ (MSO x Storage Service Units)
+ (SRB x SRB Service Units)

where CPU, IOC, MSO, and SRB are installation defined *service definition coefficients* and:

CPU Service Units = task (TCB) execution time, divided by an SRM constant which is CPU model dependent. (Refer to "Selecting Service Definition Coefficients" in Section 3). Included in the execution time is the time used by the address space while executing in cross memory mode (that is, during either secondary addressing mode or a cross memory call). This execution time is *not* counted for the address space that is the target of the cross memory reference.

I/O Service Units = measurement of individual data set I/O activity and JES spool reads and writes for all data sets associated with the address space. SRM calculates I/O service using either I/O block (EXCP) counts or device connect time (DCTI), as specified on the IOSRVC keyword in the IEAIPSxx parmlib member. If DCTI is used to calculate I/O service, operations to VIO data sets and to devices that the channel measurement facility does not time are not included in the I/O service total.

When an address space executes in cross memory mode (that is, during either secondary addressing mode or a cross memory call), the EXCP counts or the DCTI will be included in the I/O service total. This I/O service is *not* counted for the address space that is the target of the cross memory reference.

Storage Service Units = (real page frames) x (CPU service units) x 1/50, where 1/50 is a scaling factor designed to bring the storage service component in line with the CPU component. NOT included in the storage service unit calculation are the real page frames used by an address space while referencing the private virtual storage of another address space through a cross memory function (that is, through secondary addressing or a cross memory call). These frames are counted for the address space whose virtual storage is being referenced.

SRB Service Units = service request block (SRB) execution time for both local and global SRBs, divided by an SRM constant which is CPU model dependent. Included in the execution time is the time used by the address space while executing in cross memory mode (that is, during either secondary addressing mode or a cross memory call). This execution time is *not* counted for the address space that is the target of the cross memory reference.

Service Definition Coefficients

The service definition coefficients are used to assign additional weight to one type of service relative to another, allowing the installation to specify which type of resource consumption should be emphasized in the calculation of service rates. For example, an IPS may contain the following service definition coefficients:

CPU = 10.0 IOC = 5.0 MSO = 3.0 SRB = 10.0

In this case, if an address space has accumulated 100 CPU service units, 200 I/O service units and 300 storage service units, and 10 SRB service units, its total accumulated service would be:

(10x100) + (5x200) + (3x300) + (10x10) = 3000 service units

Service Rate

The rate at which service is consumed is computed by the formula:

$$\text{service rate} = \frac{\text{service}}{\text{time interval}}$$

where "service" is defined by the equation above and "time interval" is a combination of specific time periods in the life of a transaction that are defined as follows:

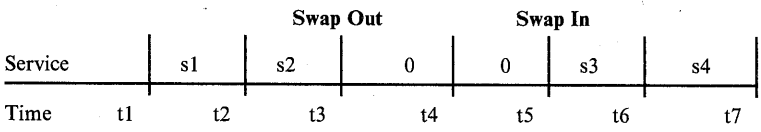
Transaction Resident Time	-	swapped in time
Transaction Out Time	-	swapped out (ready) time
Transaction Long Wait Time	-	swapped out (not ready) time
Transaction Active Time	-	Transaction Resident Time + Transaction Out Time
Transaction Elapsed Time	-	Transaction Resident Time + Transaction Out Time + Transaction Long Wait Time

"Time interval" in the service rate formula has two possible meanings:

1. For a swapped-in address space -
   time interval = last out time + current resident time.
2. For a swapped-out address space -
   time interval = last resident time + current out time.

If the out time is not factored into the time interval, the service rate is referred to as *absorption rate.*

Example of the calculation of service rate:



time t1: The transaction is starting. It has not used any service. Therefore, the service rate is 0.

time t2: The service rate equals $\frac{s1}{t2 - t1}$

The absorption rate at this time equals the service rate.

time t3: The service rate equals $\frac{s1 + s2}{t3 - t1}$

Assume the address space is now swapped out.

time t4: The service rate calculated while swapped out is

$$\frac{s1 + s2}{t4 - t1}$$

time t5: The service rate is $\frac{s1 + s2}{t5 - t1}$

Assume the address space is swapped back into storage now. The service rate is reset to 0.

time t6: The service rate calculated during this in-storage interval is $\frac{s3}{t6 - t3}$

The absorption rate now is $\frac{s3}{t6 - t5}$

time t7: The transaction completes. Its final service rate is

$$\frac{s3 + s4}{t7 - t3}$$

The total service used by the transaction is
s1 + s2 + s3 + s4.
Its total resident time is (t3 - t1) + (t7 - t5).
Its total active time is (t7 - t1).

Since the transaction did not incur any long wait time, its total elapsed time is the same as its active time.

Note: When the address space is swapped back into storage, the service rate is reset to 0, but the total service (as reported by SMF, and RMF) is not reset to 0. The total service is accumulated for the entire transaction.

Performance Objective and Workload Level

The service rate, however, is only one aspect of an address space's performance characteristics. It does not, for instance, allow consideration of the importance of a particular address space. That is, if address space A and address space B are both receiving 100 service units per second, this may represent "good" service for address space A but "bad" service for address space B.

This need for the inclusion of relative importance in the process of comparing the service received by competing address spaces leads to the more comprehensive concepts employed by SRM: *workload levels* and *performance objectives.*

A workload level specification consists of a set of positive integers of increasing magnitude, such as:

workload levels = (1,50,100)

A performance objective consists of a set of service rates and is assigned a unique identifier, such as:

objective 1, service rates = (1000,500,0)

A performance objective associates a set of service rates with a set of workload levels such that there exists, by definition, a workload level number corresponding to each specified service rate. The graph of this relationship looks as follows:

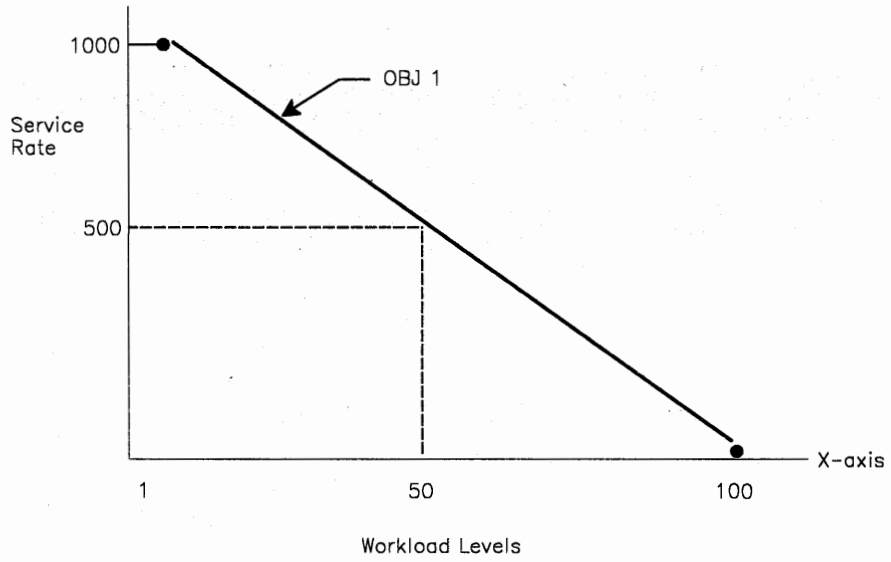


Figure 5-1. Relationship between Workload Levels and Service Rates

For every service rate on this graph, there is a corresponding workload level.

This graph is constructed by SRM from the workload level numbers and service rates specified in the IPS. The *specified* service rates are used only to construct the graph, that is, to draw the line expressing the desired relationship between *measured* service rates and workload levels. The workload levels represent a fixed common scale into which an address space's measured service rate is mapped (the purpose of this mapping is discussed later). Thus, the measured service rates are the independent variables while the workload levels are the dependent variables. (Note that the x-axis in this case represents the dependent variable, contrary to conventional usage.)

Now let us choose a second performance objective:

objective 2, service rates = (2000,1000,0)

Figure 5-2 shows this objective plotted against the same workload level numbers (there can be only one workload level specification per IPS).

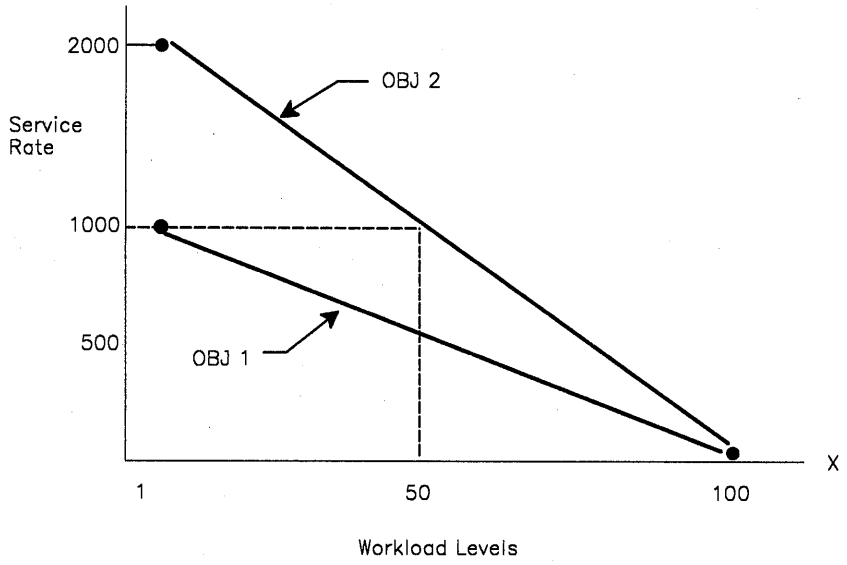


Figure 5-2. Comparing the Slopes of Two Performance Objectives

Now suppose that address space A is associated with performance objective 1 and address space B with performance objective 2, and that both address spaces, at some point in time, are receiving a service rate of 1000. For address space A, the corresponding workload level is 1, but for address space B it is 50.

These numbers, 1 and 50, are the basis for comparing the actual service rates. Thus, by associating address spaces with different performance objectives and plotting these against a common scale, the importance of each address space is factored into the comparison.

In this case, if address space A is in real storage (swapped in) and address space B is waiting to be swapped in, SRM will assign a swap recommendation value of 50 to address space B and recommend that an exchange swap be performed (see Section 1, Swapping).

Consider a third performance objective:

objective 3, service rates = (2000,2000,2000)

Since this objective does not end with a zero service rate specification, SRM will extrapolate to determine the zero point, or *cut-off level,* according to the following rules:

- The slope of the last specified segment of the graph is used and that segment is extended to the cut-off point (service rate = 0), if possible.
- The maximum cut-off point is 3/2 of the highest specified workload level number.

For performance objective 3, the cut-off level is 3/2 x 100 = 150, resulting in the following graph when combined with Figure 5-2.

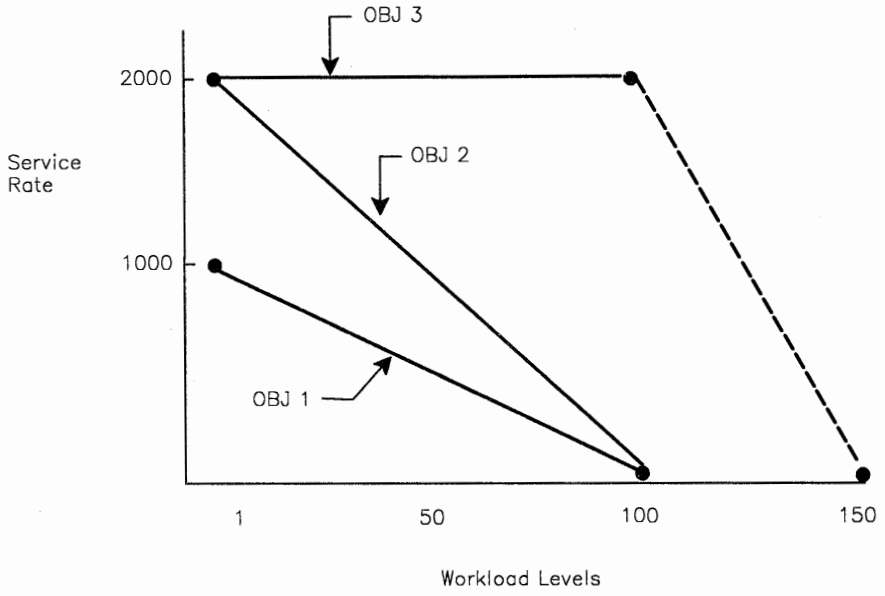


Figure 5-3. The Effect of Different Performance Objective Slopes on Service

For any service rate (< 2000) on this graph, address space C, associated with objective 3, will have a workload level greater than address spaces A and B. In effect, this insulates address space C from exchange swap considerations relative to any address space associated with objective 1 or objective 2, within the same domain. (The workload level corresponding to a service rate of 2000 for objective 3 is 100, not 1 or 50.)

Note that performance objectives are not allowed to slope upward. That is, since each succeeding service rate within a performance objective is associated with a higher workload level, no service rate may exceed a preceding one.

Let us now examine how the address spaces of a domain compete with one another on the basis of workload levels.

Consider a domain with a target MPL of 1 and two address spaces, both associated with objective 1 of Figure 5-1, and let the abbreviation Adsp1(obj1) denote that address space 1 is associated with objective 1.

Assume that Adsp1(obj1) becomes ready first and is therefore swapped in with a workload level of 100, corresponding to a service rate of 0. As it begins to accumulate service units (see definition of service), its service rate begins to increase, causing a corresponding decline in its workload level. Now the second address space, Adsp2(obj1), becomes ready and is assigned a workload level of 100, causing Adsp1(obj1) to be swapped out and Adsp2(obj1) to be swapped in. While the workload level of Adsp2(obj1) is now declining, the workload level of Adsp1(obj1) begins to increase again, since its time interval (see definition of service rate) continues to increase while it is swapped out and not accumulating service units. The result is another exchange swap.

This exchange between the two address spaces continues until one of them completes processing. The frequency of the exchange depends on the rate at which the

respective workload levels exceed one another and also, of course, on the frequency of SRM's monitoring of the situation (constant RMPTTOM, Section 6).

Let us now add an address space associated with objective 2 to this domain, Adsp3(obj2). Because of the steeper slope of objective 2, the workload level of Adsp3(obj2) decreases at a slower rate than the workload levels of Adsp1(obj1) and Adsp2(obj1), for corresponding increases in the service rate. The effect is to make Adsp3(obj2) more "important," since it will be kept in real storage for longer intervals than its rivals, assuming, of course, a similar rate of consumption of service units, and ignoring reasons for swaps other than the workload level (long waits, etc.)

If an address space associated with objective 3, Adsp4(obj3), is included in this domain, it can readily be seen that this address space will not be swapped out at all until it has completed processing, since its workload level is always greater than or equal to the workload levels of the other three address spaces (again ignoring other reasons for swaps).

Figure 5-4 illustrates these points and provides a basis for further observations.

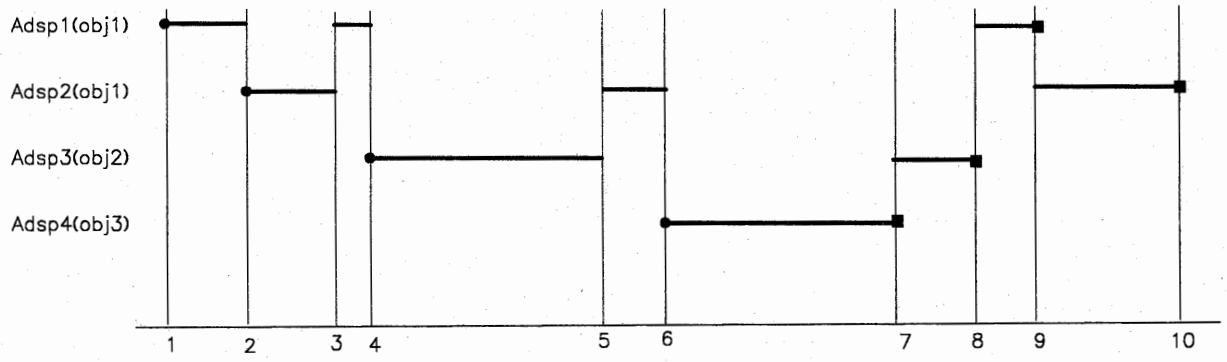


Figure 5-4. Competition for Resources Controlled by Performance Objectives

The details depicted in Figure 5-4 are arbitrary and depend entirely on such variables as the arrival times (that is, when address spaces become ready) and the amount of service required by each address space to complete processing. The concepts involved, however, will be the same.

The numbers denote the following events:

1. Adsp1(obj1) becomes ready and is swapped in with a workload level of 100.
2. Adsp2(obj1) becomes ready and is assigned a workload level of 100. Since the workload level of Adsp1(obj1) is now less than 99, Adsp1(obj1) is swapped out and Adsp2(obj1) is swapped in.
3. The workload level of Adsp2(obj1) has now declined to a level below that of Adsp1(obj1), which has increased while in the swapped out state. Therefore, Adsp1(obj1) preempts Adsp2(obj1).
4. Adsp3(obj2) becomes ready and preempts Adsp1(obj1) with a workload level of 100. Since the slope of objective 2 is steeper than that of objective 1, Adsp3(obj2) will stay swapped in for a longer interval.
5. The workload level of Adsp3(obj2) has fallen below that of Adsp2(obj1), causing it to be preempted by the latter.

6. Adsp4(obj3) becomes ready and preempts Adsp2(obj1) with a workload level of 150. It will remain in real storage until it completes because its workload level cannot fall below 100.
7. Adsp4(obj3) completes processing, allowing Adsp3(obj2) to be swapped in again, assuming that it now has the highest workload level.
8. Adsp3(obj2) completes and Adsp1(obj1) is swapped in, assuming that it has the higher workload level.
9. Adsp1(obj1) completes processing, allowing Adsp2(obj1) to be swapped in.
10. Adsp2(obj1) completes processing.

Figure 5-4 allows us to make the following observations:

- Address spaces associated with objective 3 can have a serious impact on the performance of address spaces associated with objective 1 and objective 2. This is especially true if the former represent long-running, batch-like work and the latter represent short, interactive TSO commands that require fast response times.
- If the reverse is true, that is, objective 3 is used for short TSO commands, and these arrive at a fairly regular rate, then the turnaround time for batch work may be very unpredictable.
- If batch-like work is associated with the same performance objective as short, interactive work, and objective 3 is not used, the long-running work may still be severely impacted since regularly arriving short work will constantly preempt it with a workload level of 100.
- Under the rules of competition described so far, a great deal of swapping takes place, causing a corresponding amount of overhead. This would be even more apparent if all address spaces were associated with either objective 1 or objective 2. Notice, however, that if objective 3 were the only objective used in this domain, swapping would be considerably reduced, due to the fact that much greater changes in the service rates would be required to effect similar changes in the workload levels. In other words, the frequency of swapping depends directly on the slope of the performance objective - the steeper the slope, the less frequent the incidence of swapping.

Note also that the competition for access to real storage is reduced with each increase in the target MPL of the domain, assuming a fixed maximum number of ready address spaces, and is eliminated altogether if the target MPL is equal to this maximum number.

It is often desirable to exercise a greater degree of control over the competition within a domain than is afforded by the use of performance objectives alone. Such additional control is provided through another IPS parameter, the *interval service value*.

Interval Service Value (ISV)

The purpose of the ISV is to enable an installation to control the frequency of swapping.

This is accomplished by superseding the previously discussed rules for competition within a domain for a specific interval, as follows: each time an address space is swapped in, it is assigned a service rate of zero, in effect assigning to that address space the highest possible workload level for the respective performance objective,

that is, the cut-off workload level. The ISV specifies the number of service units an address space is allowed to accumulate before the workload level corresponding to the actual service rate applies.

To illustrate the use of the ISV, consider again a domain with address spaces Adsp1(obj1), Adsp2(obj1), Adsp3(obj2) and Adsp4(obj3). Assume that each address space is associated with an ISV of 400 service units, and that the number of service units required by each address space to complete is as follows:

Adsp1(obj1) - 400 service units
Adsp2(obj1) - 600 service units
Adsp3(obj2) - 800 service units
Adsp4(obj3) - 600 service units

Figure 5-5 illustrates the effect of the ISV on the competition between these address spaces.

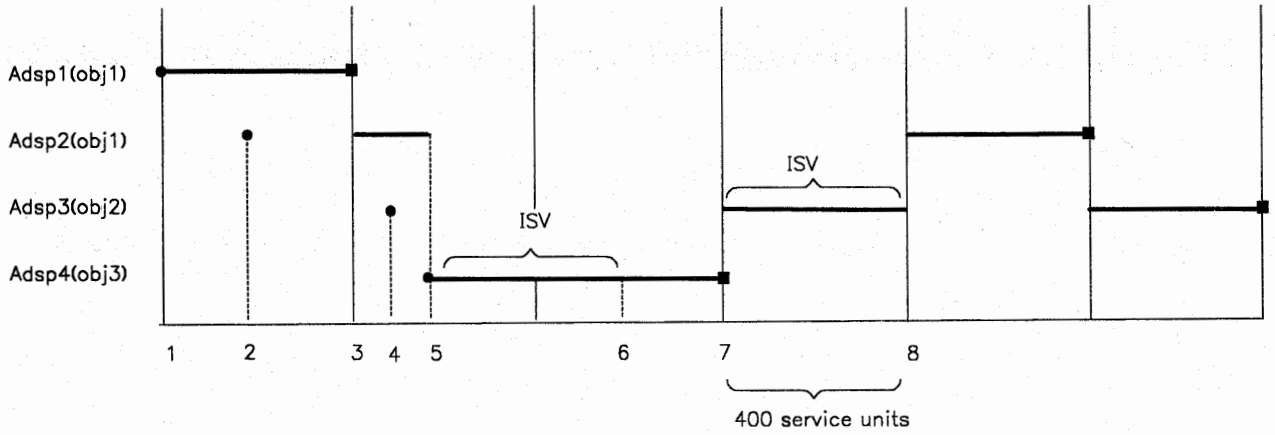


Figure 5-5. Competition for Resources Controlled by Performance Objectives and ISVs

The numbers denote the following events:

1. Adsp1(obj1) becomes ready and is swapped in. It remains in real storage for the length of its ISV and completes processing without being swapped.
2. Adsp2(obj1) becomes ready but does not preempt Adsp1(obj1) since the latter is in its ISV, that is, the two workload levels are equal.
3. Adsp1(obj1) completes and Adsp2(obj1) is swapped in.
4. Adsp3(obj2) becomes ready but cannot preempt Adsp2(obj1), since the latter is in its ISV. Notice that the only significant property of an objective, while the address space is in its ISV, is its cut-off workload level. The slope of the objective has no meaning during this interval.
5. Adsp4(obj3) becomes ready and preempts Adsp2(obj1), even though the latter is still in its ISV, due to the fact that objective 3 has a cut-off level of 150. The ISV, therefore, represents a guaranteed interval of service only when the cut-off levels of all objectives in the domain are equal.
6. Adsp4(obj3) comes out of its ISV but remains in storage since its workload level cannot decrease to less than 100.

7. Adsp4(obj3) completes processing and Adsp3(obj2) is swapped in. Notice that the service rate is computed in the usual manner while an address space is in the swapped out state. Since Adsp2(obj1) has already accumulated service units, its workload level is lower than that of Adsp3(obj2).
8. Adsp3(obj2) leaves its ISV and is preempted by Adsp2(obj1).

Figure 5-5 allows us to make the following observations:

- The frequency of swapping can be reduced by using the interval service value. Notice that Figure 5-5 shows fewer swapped in intervals than Figure 5-4.
- The ISV is most effective when the cut-off workload levels of all performance objectives in a domain differ by no more than one. Notice that Adsp4(obj3) causes Adsp2(obj1) to be swapped out while the latter is still in its ISV. The reason is the higher cut-off level of objective 3, that is 150 versus 100.
- The effectiveness of the ISV is increased when its value is increased. For example, consider the same domain without Adsp4(obj3) and assume an ISV of 800. In this case, each of the three address spaces would complete processing without exchange swaps.

The use of the ISV will be discussed again later in this section and also in Section 3.

Dispatching Priorities

Up to this point we have discussed three performance characteristics that can be associated with an address space: the domain, the performance objective, and the ISV. A fourth factor was introduced in Section 1 - the dispatching priority. Once an address space has been swapped into real storage, it competes with all other address spaces, regardless of domain, on the basis of the dispatching priorities. The higher the dispatching priority, the higher the rate at which an address space is allowed to consume resources. Thus, if an installation wishes to favor one type of work over another, it can assign a high dispatching priority in addition to favorable domain constraints, performance objectives, and interval service values.

The APG range as discussed in Section 1 is determined by the installation via the APGRNG parameter. The relationship between a job's requested dispatching priority (DPRTY) and its actual priority, as determined by SRM, is obtained through installation specified IPS parameters, assuming a job's dispatching priority falls within the APG range. In Figure 5-6 the installation wishes that the APG control the jobs that request priorities 50-6F. Through IPS defined parameters, the jobs are assigned a relative dispatching priority (DP) and a control algorithm (mean-time-to-wait or fixed) within the two APG sets.

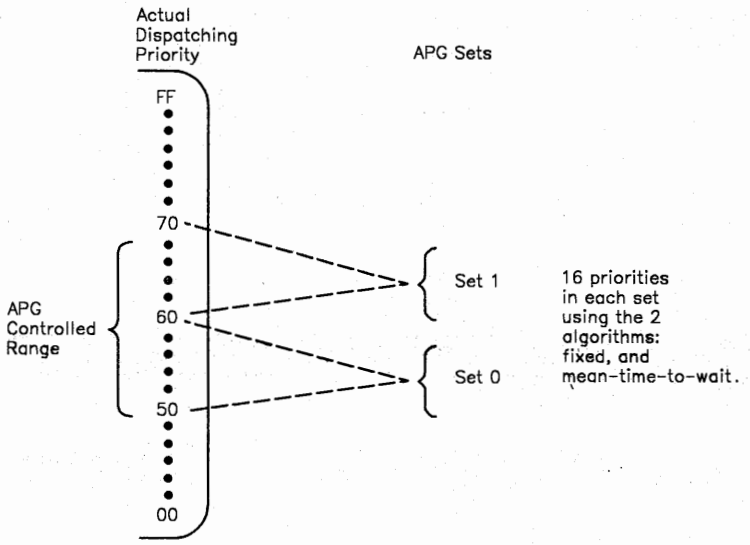


Figure 5-6. Extended Dispatching Control

The time slice function allows a job to have two different priorities within this range (DP-base priority and TSDP-time slice priority.) The amount of time the job will be at each priority is set by the installation via the TSGRP, TSPTRN, and TUNIT parameters. The TSGRP parameter defines a time slice group. Each job is associated with a time slice group by specifying this parameter within the related performance group period. The TSPTRN defines a set number of intervals (1-64) and determines which address spaces will be raised to their time slice priority during those intervals. In this way the user may control the relationship of resource utilization between jobs. For example, if two time slice groups are defined TSGRP=1 and TSGRP=2, and it is required that one group should be favored 75% of the time while the other is favored for 25% of the time, the TSPTRN parameter would be: TSPTRN=(1,1,1,2). The actual time duration represented by each interval within the TSPTRN parameter is set by the TUNIT parameter. Figure 5-7 illustrates the SRM timing parameters that are discussed in this section.

The following table relates the SRM seconds found in Figure 5-7, to wall clock time as follows:

Processor Model	SRM sec/real time sec.
3090 Model 120E	8.6957
3090 Model 120S	9.2593
3090 Model 150E	11.2360
3090 Model 150S	14.7059
3090 Model 170S	18.1818
3090 Model 180E	19.6078
3090 Model 180S	25.0000
3090 Model 200E	19.6078
3090 Model 200S	25.0000
3090 Model 280E	19.6078
3090 Model 280S	25.0000
3090 Model 300E	19.6078
3090 Model 300S	25.0000
3090 Model 400E	19.6078
3090 Model 400S	25.0000
3090 Model 500E	19.6078
3090 Model 500S	25.0000
3090 Model 600E	19.6078
3090 Model 600S	25.0000
4381 Model 91E	7.3529
4381 Model 92E	7.3529

Finally, PVLDP sets the priority of jobs in the performance group zero. Any job which is marked privileged in the program properties table will be placed in performance group zero.

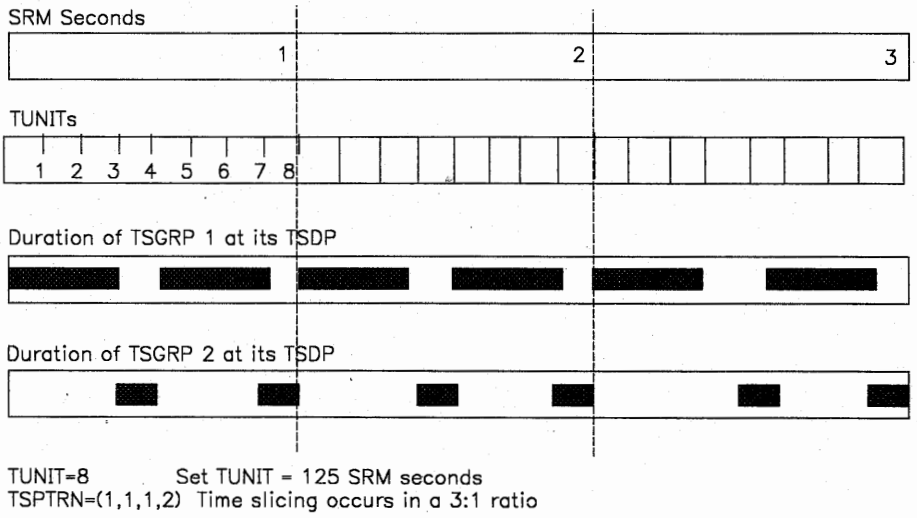


Figure 5-7. Example of SRM Timing Parameters in the IPS

I/O Priorities

If the I/O priority queuing function is requested in the IPS, I/O requests for users controlled by the APG are queued on the device according to the I/O priority of the address space. Thus, I/O priority queueing prevents important I/O from being delayed behind less important I/O. The I/O priority is by default, the same as the dispatching priority. In most cases, the default is adequate. However, in some cases it might be preferable to have an I/O priority that is higher or lower than the I/O priority assigned by default. An I/O priority for selected users can be specified in the IPS. The range of the priorities that can be specified is the same as those specified for dispatching priorities.

Notes:

1. The default I/O priority for time sliced users is the time slice dispatching priority rather than the base priority.
2. If an I/O priority is specified in a mean-time-to-wait range, it is treated as a fixed priority and assigned the lowest value in the specified mean-time-to-wait range. For example, given the following command:

   APGRNG=(6-8),DP=M0,IOP=M1

   the I/O priority would be fixed. In this case, because IOP=M1 was specified, the first digit of the dispatching priority is 7 (from the range of 6-8). If this had been for normal dispatching, the priority range would be from 70-79. Because this is an I/O dispatching priority, however, the second digit of the priority is always zero. Therefore, the fixed priority in this example is 70. (If IOP=M0 had been specified, the fixed priority would have been 60; if IOP=M2 had been specified, the fixed priority would have been 80.)

Response - Throughput Bias (RTB)

A swap recommendation based on the workload level of an address space is one of four possible factors contributing to SRM's swap decisions. The other three are recommendations from the CPU, I/O, and storage load balancing functions (see "Functions," Section 1).

An installation can specify whether one or more of the latter three recommendations should be considered by specifying the RTB parameter in the IPS. In addition, these load balancing recommendations can be given a desired weight, or importance, via the resource load balancing coefficients in the OPT (see "OPT Concepts" in this section).

The RTB has five possible values: 0, 1, I, C, or S. If the RTB is set to zero, the workload level recommendation is the sole basis for swap decisions. The effect of this choice is to emphasize an equal share of resources to address spaces of the same priority as opposed to total system throughput and responsiveness.

If the RTB is set to one, all three load balancing recommendations (CPU, I/O, and storage) are factored into the swap decisions (see "OPT Concepts"). In effect, setting RTB to one means that swap recommendations should be made in the interest of throughput, and not response time or on the basis of fair share of resources. If the RTB is set to I, C, or S, or a combination of these, the selected load balancing recommendation (I/O, CPU, or storage, respectively) is factored into the swap decisions.

TSO Response Time Control

The TSO response time parameter (RTO) limits the service given to TSO and attempts to maintain relatively consistent response time for similar TSO transactions as the system workload fluctuates. The function is effective only in a system that has more capacity than required to satisfy first period TSO transaction response times. The function can reserve the excess capacity for future applications. The function cannot improve TSO response time in a heavily loaded system.

SRM enforces the response time objective by delaying a TSO transaction after the command or other input is entered at the terminal but before the user is swapped-in. A delay is imposed only if both of the following conditions are true: a swap-in is required to process the transaction *and* the transaction was started following an input terminal wait swap. The delay is computed so that the response time of an average first period TSO transaction equals the RTO value. Smaller than average transactions have faster response times than the RTO; larger transactions have slower response times.

As the service rate increases, as when TSO is moved to a larger capacity system or when the overall system load decreases, the RTO delay is a larger percentage of the transaction response times, causing the response times for all transactions to be closer to the response time objective.

Notes:

1. Selection of a response time value does not eliminate the need to specify domains and dispatching priorities for TSO work. Careful choice of these parameters is essential to the efficiency of the system. Specifying response times complements these controls and provides consistency to the user regardless of system capacity.
2. There is no RTO delay for chained commands (commands separated by the field mark key) on a 3270 display terminal, for stacked commands on a 2741 typewriter terminal, or for commands in a CLIST.
3. If many output terminal wait conditions occur, the average first period response time will probably be less than the RTO specification.

Storage Isolation

Most applications have a critical number of pages (called its critical working set) that require a corresponding critical number of frames. If the system steals frames that reduce the application's working set below the critical number, the application's performance is impacted. In the same way, the common area (CSA, SQA, and PLPA) has a critical working set; if the system steals frames that reduce the common area's working set below this critical number, the performance of all applications that depend on the common area might be impacted.

In systems with expanded storage, the working set of an application or the common area includes the frames allocated in processor storage. Processor storage includes real and extended storage. The system disregards the target working set size when replacing pages in real storage and moving those pages to extended storage. When a page is removed from expanded storage, however, the system requires I/O to auxiliary storage to make the page available again. Thus, the system honors target working set size when migrating pages from expanded storage to auxiliary storage. The system will not migrate an expanded storage page if the result would reduce the application's or common area's storage allocation below the target working set size.

When the installation requests the storage isolation function, SRM calculates the minimum, maximum, and target working set sizes for the common area and each address space by using the installation-defined IPS parameters and the page-in rate for the common area or the address space. Through the storage isolation function, the installation can protect the working set of an application or the common area by preventing SRM's and RSM's page replacement algorithm from selecting a frame that would reduce the number of frames assigned to the application or common area to a value below the calculated target working set size. The target working set size can never be lower than the minimum working set size. SRM also uses the minimum working set size as the minimum swap-in set size. (A swap-in set consists of an address space's LSQA, fixed, and most recently-referenced pages.) If an address space has a minimum working set size of, for example, 60 frames, then, to SRM, a swapped-out address space should have at least 60 pages in its swap-in set provided that the address space had 60 or more frames allocated when it was swapped out.

Storage isolation values should be selected with care and monitored closely. It is possible to set storage isolation values that the system cannot honor. If too many address spaces are isolated and the amount of storage allocated for these address spaces equals or exceeds the total storage available, the storage available to replenish the available frame queue(s) might be insufficient. In this case, the system ignores storage isolation values and replaces pages on an LRU basis until the demand for protected storage decreases to an amount that can be contained in processor storage (real and expanded storage).

Storage isolation might be useful when:

- An application is processing a medium to light workload while batch jobs are also active. Because the application is not very active, its private area frames have a high unreferenced-interval-count (UIC), while the batch job's private area frames, being referenced at a higher rate, have a lower UIC. In this case, the common area frames are often less active than the batch job's private area frames, and therefore its frames have a higher UIC than batch. If a frame shortage occurs, the frames with the highest UIC are stolen first, and thus the common area's frames and the lightly loaded application's private area frames have a higher probability of being stolen than the batch frames.
- An application is processing a heavy workload in a storage-constrained environment where all frames appear to be recently referenced. In this case, the UIC values do not differentiate between the critical frames and the noncritical frames. Thus, all frames have an equal probability of being stolen.
- Good TSO response time for trivial transactions is important. With sufficient real storage available, you can allow larger swap-in sets in order to reduce demand page-ins for trivial transactions. The installation can use RMF Monitor I reports to determine the average swap-in set size. The RMF address space state data (ASD) report provides a detailed breakdown of swap-in set status. If the average trivial transaction has a swap-in size of 55 and the installation is using swap data sets or only local page data sets on 3380 devices, then a minimum of 60 can be set as the minimum working set size.

Note: A 60-page working set requires an allocation of five swap sets or two page groups for local page data sets on a 3380; a 55-page working set also requires an allocation of five swap sets or two page groups for local page data sets on a 3380, but half of one swap set is not used.

Because there is only one I/O seek to either write or read a swap set, the only change in swap I/O time is the time to transfer the extra five pages; this transfer causes only a very small addition to the overall swap I/O time, particularly if movable head devices are used for swapping. You can round the swap-in working set size up to a multiple of 12 by specifying the size of the minimum working set as a multiple of 12; specifying the size of the minimum working set in this way can allow full utilization of the swap sets for only a small increase in swap I/O time. The number of demand page-ins are then reduced for the trivial transactions, resulting in improved response time.

When there is no shortage of frames, no page replacement takes place, and an address space or common area can accumulate a working set larger than its target or maximum working set sizes. When a frame shortage occurs, each address space or common area that exceeds its maximum specification is examined, and enough frames are stolen to either reduce their working set sizes to their maximum, or to relieve the shortage. The frames stolen are the least-recently-referenced pages. If a shortage still exists after the common area and each address space have been reduced to their maximum working set size, the standard page replacement algorithm is used until the shortage is relieved. The system replaces pages from the common area and each address space with a current working set size that exceeds its target working set size. If the storage shortage is critical, storage isolation is ignored and frames are obtained equally from all address spaces and the common area.

Note: By default, the global resource serialization address space is given a high target working set size. Unless there is a critical shortage of available frames, this high target working set size automatically protects it from page replacement. This default can be overridden by assigning the global resource serialization address space to a nonzero performance group. (See "Control Performance Groups for System Component Address Spaces" in this chapter for more information.)

In deciding if storage isolation should be used to protect an application, consider the following:

- If storage isolation controls are used, performance can be improved for the controlled application. However, other applications can experience increased paging, which might result in decreased performance for them.
- Specifying incorrect parameter values can cause excessive demand paging, which degrades performance for the controlled application as well as the entire system.
- Protecting only the private area of an application may cause increased common area paging, which in turn can cause degraded performance for the protected application.
- Pages that are associated with the protected application or common area are not considered in determining the average system high UIC. Thus, SRM control algorithms that depend on the UIC value (such as MPL adjustment and logical swapping) might be distorted if storage isolation is used for most address spaces and the common area.
- Control of the working set size based on the page-in rate is only useful for nonswappable applications or applications that remain in real storage for intervals of about 30 to 60 seconds between swaps. Because SRM resets page-in history after a swap-in and at the start of a new transaction, controls on the page-in rate are not useful for the initial performance group period(s) defining the TSO performance objectives.

If the installation determines that storage isolation is necessary for an application, the application's use of real storage should be examined. Using RMF or SMF data and the application LOG tapes, the installation can examine the application's page-in rate, working set size, and transaction response time to establish initial parameter values. Once the values are established, performance must be evaluated to ensure that the values produce the desired results.

Domain Importance

As discussed in section 1, SRM monitors system utilization and periodically adjusts some domain's target MPL in response to installation requirements.

An installation has the ability to influence SRM's decision as to which domain's target MPL will be increased or decreased by assigning to each domain a priority. This is specified in the IPS by the CNSTR keyword by specifying minMPL, maxMPL, and either domain weight or a target control keyword.

Giving a domain a high domain importance relative to other domains implies that the work assigned to that domain should be given preference when SRM considers the system underutilized and seeks to raise some domain's target MPL to allow more address spaces in real storage.

SRM's decision is based on each domain's contention index, which is computed in one of two ways:

- if domain weights are specified -

  $$\text{contention index} = \frac{\text{average ready users x weight}}{\text{target MPL}}$$

- if target control key words are used -

  contention index = workload level.

The computation of workload level is determined by the target control keyword.

There are three target control keywords:

- AOBJ indicates that SRM will determine the workload level based on the average service rate per ready user in the respective domain, and the performance objective specified by the control keyword.
- DOBJ indicates that SRM will determine the workload level by using the total service rate of the respective domain, and the performance objective specified by the control keyword.
- FWKL indicates that SRM uses the value specified by the control keyword as the fixed workload level for that domain.

The contention index is used as follows:

1. If SRM determines that the total number of swapped-in address spaces should be increased, it selects the domain that a) has the highest contention index, b) has not yet reached its maxMPL, and c) has a target MPL less than or equal to the average number of ready users plus the maximum number of ready users in the domain; divided by 2. SRM increases that domain's target MPL by one.
2. If SRM determines that the total number of swapped-in address spaces should be decreased, it selects the domain that a) has the lowest contention index and b) has not yet reached its minMPL. SRM decreases that domain's target MPL by one.
3. If SRM determines that the total number of swapped-in address spaces should not be increased or decreased, it attempts to equalize the domain's contention indexes by periodically increasing the target MPL for the domain with the highest contention index, and decreasing the target MPL for the domain with the lowest contention index.

Performance Period

In the preceding examples, each address space has been associated with one set of performance characteristics from the time it becomes ready until processing is completed. Additional flexibility of control is gained by dividing this life span into distinct *performance periods,* and associating an address space with a different set of performance characteristics for the duration of each period. Performance periods are specified via the IPS.

The purpose of performance periods is to allow an installation to vary the performance characteristics of transactions as their execution characteristics change. For example, if a domain includes a variety of TSO users, transactions with greatly differing life spans will be competing with one another. If the majority of transactions are of short duration and these are to experience consistently good response time, it may not be satisfactory to keep one set of performance characteristics in effect for the entire life span of every transaction. By using performance periods, short transactions, for instance, can be favored over long transactions without prior knowledge of individual life spans. The following example illustrates this concept.

Consider a TSO domain with a variety of transactions whose life spans range from short to intermediate to long, where:

short is ≤ 400 service units,
intermediate is > 400 and ≤ 2000 service units,
long is > 2000 service units.

Assume that three sets of performance characteristics have been defined, as follows:

Set 1: TSO domain, objective 3, ISV=400 service units, dispatching priority = a high fixed priority, RTB=0

Set 2: TSO domain, objective 2, ISV=1000 service units, dispatching priority = a moderate fixed priority, RTB=0

Set 3: TSO domain, objective 1, ISV=1000 service units, dispatching priority = a low mean-time-to-wait priority, RTB=0

Assume also that three performance periods have been defined, such that Set 1 is in effect during Period 1, Set 2 in is effect during Period 2, and Set 3 is in effect during Period 3, for all transactions in the TSO domain, and that the length of each period is as follows:

Period 1: 400 service units
Period 2: 1600 service units
Period 3: to end of transaction

Each transaction, therefore, is associated with the following composite set of characteristics:

Set 1 for the first 400 service units, or to end of transaction
Set 2 for the next 1600 service units, or to end of transaction
Set 3 to end of transaction.

Such a composite set of performance characteristics, consisting of one or more performance periods, is called a *performance group*. A transaction is associated with a performance group via the latter's unique identifier, the *performance group number*.

For non-swappable address spaces, note that only the information specified in the first period of the associated performance group is meaningful.

Performance Group

A performance group allows an installation to associate a user's transaction(s) with a set of performance characteristics for each point in the life of the transaction(s), and thus to specify the treatment it wants a job, job step, or time-sharing session to receive at all times. Performance groups are assigned by means of the installation control specification. If an installation control specification is not in effect, batch job steps and TSO terminal sessions are associated with a performance group by the specification of a performance group number (PERFORM=nnn) on the JOB or EXEC statement, on the LOGON command, on the RESET command, or by default. (See "Installation Control Specification Concepts" later in this section).

Let us now collect many of the specifications used so far in this section into a complete IPS, using the syntax described in Section 5:

```
APGRNG=(1-10)
APGRNG=(1-10)

CPU=10.0,IOC=5.0,MSO=3.0,SRB=10.0

WKL=(1,50,100)

OBJ=1,SRV=(1000,500,0)
OBJ=2,SRV=(2000,1000,0)
OBJ=3,SRV=(2000,2000,2000)

DMN=1,CNSTR=(1,2),FWKL=1
DMN=2,CNSTR=(4,8),FWKL=10

PGN=1,(DMN=1,OBJ=1,ISV=400, ...)

PGN=2,(DMN=1,OBJ=2,ISV=400, ...)

PGN=3,(DMN=1,OBJ=3,ISV=400, ...)

PGN=4,(DMN=2,OBJ=3,ISV=400,DP=F30,RTB=0,DUR=400)
      (DMN=2,OBJ=2,ISV=1000,DP=F20,RTB=0,DUR=1600)
      (DMN=2,OBJ=1,ISV=1000,DP=M0,RTB=0)
```

where:

Domain 1 (DMN1) is the domain used for the examples illustrated by Figure 5-4 and Figure 5-5;

Domain 2 is the TSO domain described under "Performance Period";

Performance groups 1, 2 and 3 (PGN1,..) represent the partial sets of characteristics illustrated by Figure 5-5;

Performance group 4 is the composite set described under "Performance Period."

We now return to the discussion of the TSO domain and performance group PGN4. The following observations can be made about transactions competing in this domain:

- During Period 1, all transactions are associated with performance objective OBJ3, which represents relatively "good" service (see Figure 5-3), and a high fixed APG priority. An ISV of 400 ensures that no exchange swapping takes place at all during this period, since this value is equal to the duration of the period. Note again that each specification is in effect only for the duration of the respective period.
- Transactions requiring more than 400 service units are then switched to the less favorable performance objective OBJ2 (a lower cut-off level), and a lower fixed dispatching priority.
- Transactions requiring more than 2000 service units (DUR=400 + DUR=1600) are then switched to objective OBJ1, and a low mean-time-to-wait APG priority (see "Functions," Section 1).

- Under these conditions, intermediate and long commands (transactions completing during Period 2 and Period 3, respectively), will probably exhibit poor response times during periods of high terminal activity. Notice that transactions in Period 1 will preempt those in Period 2 and Period 3 even when the latter are in their ISV interval (see Figure 5-5). To prevent swapping during ISV intervals, the cut-off workload levels of the performance objectives must be equal, as in the following performance group (PGN4 without objective OBJ3):

```
PGN=5,(DMN=2,OBJ=2,....)
      (DMN=2,OBJ=2,....)
      (DMN=2,OBJ=1,....)
```

Under the conditions of performance group PGN5, the response time of short commands may become erratic since transactions in Period 2 and Period 3 now have a much higher chance of gaining access to real storage. This problem may be alleviated somewhat by decreasing the ISV for Period 3 to 1000 service units. Another solution would be to remove transactions entering Period 3 from the domain, as follows:

```
PGN=6,(DMN=2,....)
      (DMN=2,....)
      (DMN=1,....)
```

The purpose of the examples in this section is to illustrate concepts, not to serve as recommendations.

The SET IPS Command

The operator can select a new IPS (that is, indicate that the system is to run under the control of an alternate IEAIPSxx member) between IPLs by issuing the SET IPS command. When the SET IPS command is processed, any ongoing transactions are associated with the first performance group period of the new performance group (that is, the previous transaction is ended, and a new transaction is started, before processing continues). If the performance group with which ongoing transactions were previously associated is not defined in the new IPS, they are associated with the appropriate default performance group (1 for batch users, 2 for TSO users). SET command processing also notifies RMF to terminate and reinstate its workload reporting. Thus, RMF synchronizes its reports with the time of the SET and provides consistent data.

The operator can also change the performance group of an ongoing job or TSO session by issuing the RESET job name, PERFORM=nnnn command. The RESET command changes only the control performance group for the job or session; it does not change any of the reporting performance groups. For more information on assigning control and report performance groups, see the section "Installation Control Specification Concepts." (Refer to *Operations: System Commands* for detailed syntax information on both the SET and RESET commands).

Installation Control Specification Concepts

The installation control specification, located in the IEAICSxx member of SYS1.PARMLIB, is a central place for assigning performance groups to units of work called transactions. Through the installation control specification, an installation can control performance group assignment for selected subsystems. For information on performance group assignments for subsystems not included in the installation control specification, see "Performance Group" earlier in this section.

The installation control specification replaces the PERFORM parameter in job entry subsystems, job control language (JCL), LOGON commands, and the TSO user attribute data set (UADS). Until an installation writes an IEAICSxx member and puts it in effect, the PERFORM parameter is used for performance group assignment.

There are two types of performance groups that can be assigned:

- Control performance groups - to control transactions according to a set of performance characteristics defined in the IPS. There can be only one control performance group for a transaction at any one time.
- Report performance groups - to report statistics on the transactions through the RMF workload activity report. (Control performance groups can also serve a reporting function). Depending on the number of ways statistics are collected, there can be more than one report performance group for a transaction. Report performance groups for a transaction are obtained solely from the installation control specification.

The control performance group for a transaction is obtained from two sources: the installation control specification and the PERFORM parameter value specified by the user or installation.

SRM uses the values in the installation control specification to verify and, if necessary, override any PERFORM value. The PERFORM value is used as the control performance group only when there is no subsystem entry in the installation control specification applicable to the transaction or if the PERFORM value is allowed by the installation control specification. (The value in the PERFORM parameter must be specified as a PGN or OPGN in the installation control specification.)

In any case, the control performance group must be defined in the IPS. Otherwise, the system default is assigned.

Subsystem Sections

The installation control specification consists of multiple sections, one for each subsystem whose transactions are to be assigned performance group numbers (PGNs). Each subsystem section contains entries that identify various groups of transactions according to their transaction name, userid, or transaction class. Each entry specifies a control or report performance group number, or both.

A control or report PGN can be specified for the subsystem as a whole. This allows an installation to specify a default control PGN or to specify a report PGN to accumulate statistics for the entire subsystem.

Note: The installation control specification does not assign control PGNs to initiators or privileged jobs. These are always controlled in performance group zero, a system-defined performance group.

The following subsystems can be included in an installation control specification. Only the first three subsystems are subject to control PGN assignment, because their transactions are under direct SRM control. Report PGNs are valid for all the subsystems.

- STC subsystem (system-defined). The transactions for this subsystem include all work initiated by the START and MOUNT commands. STC also includes the

system component address spaces such as the global resource serialization and PC/AUTH address spaces.

- TSO subsystem (system-defined). The transactions for this subsystem include all commands issued from foreground TSO sessions. Reporting for TSO command is active only if TSO Extensions is installed.
- Job entry subsystem (usually JES2 or JES3). The transactions for this subsystem include all jobs that it initiates.
- Other interactive subsystems (such as VSPC or a user-written subsystem). The transactions for these subsystems are all those reported to SRM through the transaction reporting interface.

Note: Refer to the current documentation for an IBM subsystem to determine whether or not it uses the reporting interface. The reporting interface consists of the SYSEVENT macro instruction, which is used to pass transaction information to SRM. For information on using the SYSEVENT reporting macro instruction, see *SPL: System Macros and Facilities*

The following is a sample installation control specification and a portion of its corresponding IPS:

IEAICSxx

```
SUBSYS=STC, PGN=1
SUBSYS=TSO, PGN=2
SUBSYS=JES2, PGN=3
SUBSYS=VSPC, RPGN=4
```

IEAIPSxx

```
PGN=1,(DMN=1,DP=M1,...)
PGN=2,(DMN=2,DP=F14,...)
PGN=3,(DMN=3,DP=M0,...)
```

The sample IEAICSxx member has four subsystem sections; each specifies a performance group that applies to the subsystem as a whole. All started tasks, including VSPC and JES2, are controlled and reported on in performance group 1; all TSO users in performance group 2; and all JES2 jobs in performance group 3. In addition, VSPC transactions are reported on in performance group 4. In this example, VSPC is reported on in two ways:

- SRM views the VSPC address space as a transaction. All resources used by the address space are reported in PGN 1 as a started task.
- The individual transactions processed by VSPC and identified to SRM through the transaction reporting interface, are reported in PGN 4.

Note that performance group 4, because it is used only for reporting, is not defined in the IPS.

Transaction Entries

Within each subsystem section, the installation control specification entries can identify transactions in four ways:

- By transaction name (TRXNAME keyword)
- By userid (USERID keyword)
- By transaction class (TRXCLASS keyword)
- By accounting information (ACCTINFO keyword)

Not all keywords are meaningful for each subsystem. The following table defines the installation specification keywords and indicates whether the keywords are meaningful, and if so, whether SRM controls or reports on the transaction defined by the keyword. Report performance groups are valid for all entries.

Subsystem	Keyword	Control PGN (PGN)	Report PGN (RPGN)
TSO	TRXNAME - The TSO command name[1]	No	Yes
	USERID - The userid specified at LOGON	Yes	Yes
	TRXCLASS - N/A	N/A	N/A
	ACCTINFO -Accounting information	Yes	Yes
STC	TRXNAME - The name as specified on the start command or the name of the system address space (these names are given in a following section, "Control Performance Groups for System Component Address Spaces").	Yes	Yes
	USERID - N/A	N/A	N/A
	TRXCLASS - N/A	N/A	N/A
	ACCTINFO -N/A	N/A	N/A
JES2[2]	TRXNAME - The jobname of the JES2-initiated job	Yes	Yes
	USERID - The userid specified on the JOB card via the RACF USER keyword	Yes	Yes
	TRXCLASS - The job class used for work selection	Yes	Yes
	ACCTINFO - Accounting Information	Yes	Yes
JES3[2]	TRXNAME - The jobname of the JES3-initiated job	Yes	Yes
	USERID - The userid specified on the JOB card via the RACF USER keyword	Yes	Yes
	TRXCLASS - The job class used for work selection	Yes	Yes
	ACCTINFO - Accounting Information	Yes	Yes
ANY[3]	TRXNAME - The transaction name (usually command name)	No	Yes
	USERID - The userid specified at LOGON.	No	Yes
	TRXCLASS - The transaction class	No	Yes
	ACCTINFO - Accounting Information	No	Yes

[1]The TSO LOGON command is not reported on through the installation control specification unless LOGON is used to terminate a TSO session. TSO subcommands are reported under the primary command name. Commands within a TSO CLIST are reported on under the EXEC command. Reporting for TSO commands is active only if TSO Extensions is installed.

[2]The job entry subsystem name is the primary or alternate subsystem name specified on the SCHEDULR sysgen macro, or in the IEFSSNxx parmlib member, or in the IEFJSSNT load module in SYS1.LINKLIB. This table and the remainder of this section assume that the name is either JES2 or JES3.

[3]For the purpose of illustration, this section assumes that a hypothetical user-written subsystem ANY, invokes SRM's reporting interface. This interactive subsystem is included only as an example of a subsystem that could use SRM's reporting interface. Refer to the current documentation for a IBM or user-written subsystem to determine if the subsystem uses the interface, and, if it does, what the subsystem name is and which keywords are valid.

The following sample installation control specification and a portion of its corresponding IPS illustrate the use of transaction entries within a subsystem:

IEAICSxx

```
MASK=*
SUBSYS=JES2,PGN=3
   TRXNAME=PAYROLL,PGN=30
   USERID=D58RPM1,PGN=31
   TRXCLASS=AB,RPGN=32
   ACCTINFO=D58***(8),RPGN=33
```

IEAIPSxx

```
PGN=3(DMN=3,...)
PGN=30,(DMN=3,...)
PGN=31,(DMN=3,...)
```

PAYROLL is controlled in performance group 30. All other jobs that specify userid D58RPM1 in their JCL are controlled in performance group 31. All remaining JES2 jobs are controlled in performance group 3. In addition, all statistics for CLASS=AB jobs (even PAYROLL if it is CLASS=AB) are reported on in performance group 32. All jobs with the problem numbers starting with D58 (in the eighth position) followed by three characters will be assigned to report performance group 33.

Assignment of Control Performance Groups

The previous example shows that there is a hierarchy used when searching for a control performance group. This hierarchy is necessary because it is possible for a single transaction to match several entries in the installation control specification. However, multiple control performance groups cannot be honored for the transaction; only one control performance group can be assigned. **Note:** This hierarchy is not meaningful for report performance groups because multiple RPGNs can be assigned to a single transaction. One report PGN is possible for each installation control specification entry that matches the transaction.

The following diagram describes SRM's search through the installation control specification to determine the control performance group for a transaction:

SUBSYS The search first locates the appropriate subsystem section in the installation control specification. If the subsystem is not found, control performance group assignment is made by means of the PERFORM parameter. If the subsystem is found, the search continues with TRXNAME.

TRXNAME If the transaction name matches a TRXNAME entry and the entry specified a control PGN, this PGN is assigned to the transaction. Otherwise, the search continues with USERID.

Note: For the global resource serialization address space (SUBSYS=STC, TRXNAME=GRS), a default of PGN 0 is assigned if no PGN is specified, and the search ends.

USERID If the transaction user id matches a USERID entry and the entry specified a control PGN, this PGN is assigned to the transaction. Otherwise, the search continues with TRXCLASS.

TRXCLASS If the transaction class matches a TRXCLASS entry and the entry specifies a control PGN, this PGN is assigned to the transaction. Otherwise, the search continues with ACCTINFO.

ACCTINFO If the transaction class matches an ACCTINFO entry and the entry specifies a control PGN, this PGN is assigned to the transaction. Otherwise, the search continues with the SUBSYS entry.

SUBSYS If the subsystem entry specifies a control PGN, this PGN is assigned to the transaction. Otherwise, a default is assigned as follows:

- STC and job entry subsystem = performance group 1
- TSO subsystems = performance group 2

Note: These defaults are the same as the PERFORM parameter defaults. They are also the defaults used when the control PGN assigned in the installation control specification is not defined in the IPS.

Control Performance Groups for System Component Address Spaces

System component address spaces can be assigned control performance groups under the STC subsystem name using the TRXNAME keyword. The system assigns the following names to the system component address spaces:

PCAUTH	Cross memory authorization
TRACE	System trace
GRS	Global resource serialization
DUMPSRV	Dumping services
CONSOLE	Communications task
ALLOCAS	Allocation data areas and routines
SMF	System management facilities
LLA	LNKLST lookaside
JES3AUX	JES3 data areas and programs

The only purpose in assigning control PGNs to these address spaces is to specify storage isolation parameters for them. Swapping parameters are irrelevant because system component address spaces (except for DUMPSRV) are nonswappable. Dispatching priority parameters are also unimportant because system component address spaces are accessed primarily through cross memory functions; the dispatching priority is that of the home address space. However, note that during initialization and refresh of the LLA address space, a high dispatching priority will reduce the time required to initialize LLA. Service statistics are not useful because processor time and EXCP counts accumulated in cross memory mode are counted as service to the home address space and not to the target address space.

The default control PGN for the system address spaces is the normal STC subsystem default except for the global resource serialization address space and the DUMPSRV address space; their default is PGN 0. SRM gives the global resource serialization address space special storage isolation parameters that protect its pages from being stolen except during a critical shortage of available frames. Although not recommended, an installation can choose to lower the global resource serialization address space's protection by assigning it to a nonzero control PGN in the installation control specification and defining this control PGN in the IPS with the selected storage isolation parameters. Note that lowering of the storage protection for the global resource serialization address space could cause global resource serialization to suffer page faults in critical serialization paths, thus impacting performance. Also the system assigns the highest dispatching priority (X'FF') to the global resource serialization address space to guarantee that it gets good service when it does execute as the home address space. This dispatching priority remains at X'FF' even when you assign a nonzero control PGN to the global resource serialization address space.

Optional Control Performance Groups

An installation can include a subsystem in the installation control specification and at the same time allow the PERFORM parameter to be used for selected transactions. Thus, a subsystem can be controlled primarily through the installation control specification while a subset of its transactions are controlled through the PERFORM parameter. (For more information on the PERFORM parameter, see "The PERFORM Parameter" in Section 4).

Any IEAICSxx entry that specifies a control PGN can also specify one or more optional control performance groups. A transaction that matches this entry (subject

to the searching order) can specify one of the optional performance groups on the PERFORM parameter. For example, assume the following IEAICSxx entry:

```
SUBSYS=JES3,PGN=1,OPGN=(11,12,13)
 TRXCLASS=X,PGN=4,OPGN=14
```

Class X jobs are allowed to specify PERFORM = 14 in their JCL. Class X jobs that omit the PERFORM parameter or specify a PGN other than 14 are controlled in performance group 4. JES3 jobs in any class other than X are allowed to specify PERFORM = 11, 12, or 13. Otherwise, JES3 jobs are controlled in performance group 1. Class X jobs are not allowed to specify PERFORM = 11, 12, or 13.

Note: If TSO Extensions is not installed, the optional performance groups for TSO must be specified in both the installation control specification and in the user attribute data set (UADS). If they are not specified in the UADS, the PERFORM parameter is ignored at LOGON, causing the normal installation control specification control PGN to be assigned. With TSO Extensions installed, the optional performance groups need only be specified in the installation control specification.

Report Performance Groups

Transaction data is always accumulated in the control performance group. However, report performance groups (RPGNs) can be used to obtain additional reports at various levels of detail within a subsystem. Report performance groups are primarily intended for use with interactive subsystems that use SRM's reporting interface and for TSO command reporting. They can also be used with any subsystem that is defined in the installation control specification. Report performance group numbers are valid with any of the four entry types (SUBSYS, TRXNAME, USERID, TRXCLASS, and ACCTINFO).

Report performance groups can accumulate the following types of data: the number of ended transactions and average response times. This data can be accumulated with or without service usage data. For TSO, STC, JES2, or JES3 subsystems, report performance groups contain response time and service usage data. For all other subsystems, refer to the current documentation to determine whether or not service usage data is passed to the reporting interface.

A report performance group accumulates data for all transactions that match the installation control specification entry. Thus, while a single transaction can have only one performance group, it can be reported in as many as four report performance groups, one for each type of entry.

To collect data through report performance groups, RMF workload activity reporting must be active. To prevent double counting in the same report performance group, the following restrictions apply:

- In an installation control specification, a defined RPGN cannot be the same as a control PGN. (For example, a RPGN can never be 1 or 2, which are the control PGN defaults).
- Within a subsystem, the same RPGN cannot be specified for more than one entry type. For example, the same RPGN could not be used to collect data for both a TSO userid and a TSO command. However, the same RPGN can be specified for multiple entries of the same type, for example, multiple VSPC userids. Also, the same RPGN can be specified in more than one subsystem section so that the report spans several subsystems. For example, the TCAM started task can be included in a total TSO subsystem report.

- A report performance group should not be defined in the IPS. If it is, a transaction might be assigned this performance group as a control performance group through the PERFORM parameter, causing unwanted data to be collected in the group. This problem can occur if the installation control specification omits one or more of the subsystems (STC, TSO, or job entry). Omitting the subsystem allows the PERFORM parameter to determine the control performance group. The problem could also occur through a RESET command because the command overrides the installation control specification.

The following example illustrates how control and report performance groups are used.

```
SUBSYS=JES2,PGN=10,RPGN=100
 TRXNAME=SORT,PGN=11
 TRXCLASS=A,RPGN=110
```

This installation control specification designates the following:

- The JES2 job SORT is controlled and reported in PGN 11.
- All other JES2 jobs are controlled and reported in PGN 10.
- All CLASS=A jobs, including SORT (if it is CLASS=A), are reported in RPGN 110.
- In addition, all JES2 jobs are summarized in RPGN 100.

Note: CLASS=A jobs are reported in RPGN 100, RPGN 110, and either PGN 10 or PGN 11.

Substring Notation

In the installation control specification, transactions can optionally be identified by a common substring of characters rather than by a full name. By using substrings, an installation can, for example, request RMF reports on TSO by department or some other user grouping, assuming conventions are followed for generating user ids.

Example 1: Assume that all TSO userids for department D09 begin with the character "D09." The following IEAICSxx entry could be written:

```
SUBSYS=TSO,PGN=2
 USERID=D09(1),RPGN=20
```

According to this entry, all TSO users are controlled in performance group 2. In addition, statistics for department D09 users are collected in performance group 20. The number following the characters 'D09' in the IEAICSxx entry indicates that the substring begins in column 1 of the user identifier.

Example 2: Substrings can also be used to specify TSO commands and their aliases.

```
SUBSYS=TSO,PGN=2
 TRXNAME=EX(1),RPGN=21
```

In this example, statistics for all TSO commands that start with an 'EX', namely EXEC and its alias EX are collected in performance group 21. If TRXNAME=EXEC had been specified, any EXEC command entered at the terminal as an 'EX' would not be reported on in PGN 21.

Example 3: Substrings can begin at any position in a name.

```
SUBSYS=JES2,PGN=1
 TRXNAME=LKED(5),PGN=10
```

All JES2 jobs with job names that contain 'LKED' starting in the fifth position are controlled in performance group 10.

Searching Order for Substrings and Masking

When assigning a control or report PGN, the installation control specification produces only one match within a particular entry type. That is, a transaction can match at most one SUBSYS, one TRXNAME, one USERID, one TRXCLASS and one ACCTINFO entry. If the transaction matches an entire entry, unmasked and non-substring, this is the first entry to be used. The next criteria is based on the number of specified character; a specified character is any non-masked character including blanks. If two entries have the same number of specified characters, then the first one specified in the ICS is used.

Example 1: The following IEAICSxx entry illustrates the substring searching order.

```
MASK=*
SUBSYS=TSO,PGN=2
 USERID=D0(1),PGN=3
 USERID=GES(4),PGN=4
 USERID=D09(1),PGN=5
 USERID=D09GES1,PGN=6
 USERID=D09****, PGN=7
```

The following assignments are made:

Userid D09GES1 is controlled in PGN 6
Userid D09GES2 is controlled in PGN 4
Userid D09BRP1 is controlled in PGN 5
Userid D09JMB1 is controlled in PGN 7
Userid D08SFS1 is controlled in PGN 3

Example 2: To report a TSO command and its alias in the same performance group, specify the command as a substring equal to its alias with a position number of 1.

However, there are some exceptions to this general procedure. For instance, if EDIT commands are required, TRXNAME=E(1) also counts EXECs unless TRXNAME=EX(1) also appears in the installation control specification. The same is true for R(1) when counting RUN and RENAME commands.

The following RPGN assignments cause EDIT, EXEC, RUN, and RENAME commands to be reported separately.

```
SUBSYS=TSO,PGN=2
  TRXNAME=E(1),RPGN=21     /*EDIT,E                    */
  TRXNAME=EX(1),RPGN=22    /* EXEC,EX                  */
  TRXNAME=R(1),RPGN=23     /* RUN,R                    */
  TRXNAME=RE(1),RPGN=24    /* RENAME,RE                */
```

Note: An installation that defines its own TSO commands with aliases that are not simple substrings of the full name must include additional entries for the aliases. Otherwise, the aliases are not reported in the same performance group as the full name.

The SET ICS Command

The operator can use the SET command between IPLs to change to another IEAICSxx parmlib member. If the system was IPLed without an installation control specification, SET can place one into effect. Once an installation control specification is in effect, the operator can remove the parmlib controls by issuing a SET command specifying an empty IEAICSxx parmlib member.

When a SET occurs, all transactions in the system are changed to the new parmlib controls. If the new parmlib member changes a transaction's control performance group, that transaction ends and a new one begins in the first period of the new performance group. SET command processing also notifies RMF to terminate and reinstate its workload reporting. Thus, RMF synchronizes its reports with the time of the SET and provides consistent data.

Care must be taken when using the SET command to change to an installation control specification that omits a previously-specified subsystem. Any transaction in that subsystem that starts after the SET is issued is assigned a control performance group according to its JCL or LOGON PERFORM parameter value. However, a transaction that is active at the time of the SET retains its last IEAICSxx-assigned control performance group. The transaction is not assigned its original PERFORM value.

For syntax information on the SET ICS command, see *Operations: System Commands.*

Installation Control Specification Examples

The following section contains two examples of an installation control specification and the related portions of its corresponding IPS.

Example 1:

IEAICSxx

```
SUBSYS=TSO,PGN=2
SUBSYS=STC,PGN=5
  TRXNAME=IEEVMPCR,PGN=6
SUBSYS=JES2,PGN=1
```

IEAIPSxx

```
PGN=2,(DMN=,DP=,...)
PGN=5,(DMN=,DP=,...)
PGN=6,(DMN=,DP=,...)
PGN=1,(DMN=,DP=,...)
```

This example shows the assignment of control performance groups based on the type of work (TSO, started task, mount, batch). Specifying a PGN for TSO allows for the automatic override of a PERFORM parameter on the LOGON command. Specifying the TSO subsystem in that installation control specification eliminates the need to check for optional PGNs in the UADS data set if TSO Extensions is installed. All TSO users are assigned to performance group 2.

Assigning unique performance groups for started tasks and MOUNT commands allows measurement and control of the work without having to modify JCL in SYS1.PROCLIB. If PGN 5 and PGN 6 are defined in the current IPS, all started tasks including the JES2 address space, are assigned to performance group 5 and mounts to performance group 6.

All work initiated by JES2 is controlled and reported on in performance group 1.

Example 2:

IEAICSxx	**IEAIPSxx**
SUBSYS=ANY,RPGN=12	
TRXNAME=GETNEXT,RPGN=13	
TRXNAME=INQUIRE,RPGN=14	
TRXNAME=REPORT,RPGN=15	
TRXCLASS=255,RPGN=16	
SUBSYS=VSPC,RPGN=31	
SUBSYS=TSO,PGN=40	PGN=40,(DMN=,DP=,...)
USERID=CONTROL,PGN=50	PGN=50,(DMN=,DP=,...)
USERID=TT(1),PGN=60,OPGN=(61,62,63)	PGN=60,(DMN=,DP=,...)
	PGN=61,(DMN=,DP=,...)
	PGN=62,(DMN=,DP=,...)
	PGN=63,(DMN=,DP=,...)
USERID=SS(1),PGN=70,OPGN=(71,72,73)	PGN=70,(DMN=,DP=,...)
	PGN=71,(DMN=,DP=,...)
	PGN=72,(DMN=,DP=,...)
	PGN=73,(DMN=,DP=,...)
USERID=RR(1),PGN=80,OPGN=(81,82,83)	PGN=80,(DMN=,DP=,...)
	PGN=81,(DMN=,DP=,...)
	PGN=82,(DMN=,DP=,...)
	PGN=83,(DMN=,DP=,...)
TRXNAME=TIME,RPGN=100	
TRXNAME=LISTC(1),RPGN=110	
TRXNAME=ALLOC(1),RPGN=130	
SUBSYS=STC,PGN=1	PGN=1,(DMN=,DP=,...)
TRXNAME=ANY,PGN=5	PGN=5,(DMN=,DP=,...)
TRXNAME=IMSCR,PGN=10	PGN=10,(DMN=,DP=,...)
TRXNAME=MPR(1),PGN=11	PGN=11,(DMN=,DP=,...)
TRXNAME=VSPC,PGN=30	PGN=30,(DMN=,DP=,...)
TRXNAME=IEEVMPCR,PGN=1	
SUBSYS=JES3	
TRXNAME=PAYROLL,PGN=90	PGN=90,(DMN=,DP=,...)
TRXNAME=MRBIG,PGN=90	
TRXCLASS=A,PGN=91	PGN=91,(DMN=,DP=,...)
TRXCLASS=S,PGN=92	PGN=92,(DMN=,DP=,...)
TRXCLASS=T,PGN=93	PGN=93,(DMN=,DP=,...)
ACCTINFO=D58(8),PGN=94	PGN=94,DMN=,DP=,...)

This example shows the assignment of control and report PGNs based on TRXNAME, USERID, TRXCLASS, ACCTINFO, and type of work. This example also illustrates the use of OPGN and substring syntax.

SUBSYSTEM ANY

ANY is a hypothetical user-written subsystem that invokes SRM's reporting interface. Reporting is assigned based on the transactions' names and classes. Response times for transactions with names of GETNEXT, INQUIRE and REPORT are accumulated in performance groups 13, 14, and 15, respectively. All transactions, including the three names above, that belong to class 255 are reported in performance group 16. Finally, response time for all transactions, regardless of name or class is accumulated in performance group 12. Because these performance groups are used for reporting and not for control, the IPS does not define performance groups 12-16.

The assignment of transactions to report performance groups and the fact that the subsystem uses the transaction reporting interface allow SRM to collect performance data on the transactions and to report this data through RMF.

SUBSYSTEM VSPC

All VSPC commands are reported in performance group 31. The assignment of VSPC transactions to performance groups and the fact that VSPC uses the transaction reporting interface to allow SRM to collect performance data on the transactions and to report this data through RMF.

SUBSYSTEM TSO

The user whose userid is "CONTROL" is assigned the domain and dispatching controls defined by PGN 50. All users whose userid begins with RR and who do not request a performance group are assigned to PGN 80. However, they can request PGNs 81, 82, or 83 at LOGON time. All users whose userid begins with SS who do not request a performance group are assigned to PGN 70. However, they can request PGNs 71, 72, or 73. All users whose userid begins with TT who do not request a performance group are assigned to PGN 60. However, they can request PGNs 61, 62, or 63. Note that RR, SS, and TT specify the first two characters of the USERID. All other TSO users are controlled by performance group 40. The assignment of TSO users to installation specified performance groups ensures that the execution control parameters assigned to the users meet the installation's requirements.

Notes:

1. If TSO Extensions is not installed, the optional performance groups must be specified in both IEAICSxx and the user attribute data set (UADS). If they are not listed in the UADS, the PERFORM parameter is ignored at LOGON, causing the normal installation control specification control PGN to be assigned. When TSO Extensions is installed, the optional performance group need only be specified in IEAICSxx.

   In addition to the usual transaction statistics recorded in the control performance groups, all transactions for the TSO commands TIME, LISTCAT, and ALLOCATE will have service statistics accumulated in performance groups 100, 110, and 130, respectively. Again, note that the IPS does not define these report performance groups.

2. Unless TSO Extensions is installed, the installation cannot report on TSO commands by name.

SUBSYSTEM STC

The subsystem ANY, which is a started task, is assigned to control PGN 5. The TRXNAME parameter is also used to assign the PGNs 10 and 30 to the IMS control region and VSPC, respectively. The message processing regions (MPRs) have unique jobnames MPR1, MPR2, and so forth, and they are assigned to PGN 11. All mounts are assigned to performance group 1. Because performance group 1 is the default performance group for started tasks, the mount TRXNAME entry in the ICS can be deleted with no change in control performance group assignments.

The assignment of started tasks to installation specified performance groups ensures that the execution control parameters assigned to the started tasks meet the installation's requirements.

SUBSYSTEM JES3

If initiated by JES3, the PAYROLL and MRBIG jobs are assigned to performance group 90. Jobs belonging to JES3 classes A, S, or T are assigned performance groups 91, 92, or 93, respectively. All other batch jobs are assigned to performance group 1, the default specification for non-TSO work. The assignment of batch jobs to installation specified performance groups ensures that the execution control parameters assigned to the jobs meet the installation's requirements.

OPT Concepts

The parameters discussed in this part are explained in "IEAOPTxx Parameters."

Resource Load Balancing Coefficients

The purpose of the resource load balancing coefficients is to enable an installation to assign the desired weight to the swap recommendation values produced by the CPU, I/O, and storage load balancing functions (see "Resource Use Functions," Section 1). If the response-throughput bias (RTB) in the IPS is not zero, SRM combines the specified weighted values with the workload level recommendations to determine the full swap recommendation value of an address space. Setting RTB equal to one causes all three load balancing values to be used. Any combination of load balancing values can be used by setting RTB to one or more of the values C, I, or S, for the CPU, I/O, and storage load balancers, respectively. This combined value is the basis for swap decisions. It is computed by the following formula:

swap recommendation value =
workload level recommendation
\+ (CPU coefficient x CPU load balancing recommendation)
\+ (I/O coefficient x I/O load balancing recommendation)
\+ (Storage coefficient x storage load balancing recommendation)

Note: The CPU, I/O, and storage swap recommendation values produced by the load balancing functions can be either positive or negative values.

Setting any coefficient to zero eliminates that recommendation from consideration and allows SRM to bypass the algorithms that control that particular load balancer. For this reason, it is advisable to set the load balancer coefficient to zero if the installation does not wish to use that function. Setting all the coefficients to zero is equivalent to a specification of RTB=0 in the IPS.

The values generated by the load balancing functions depend on the total CPU, I/O, and storage imbalance and on the percentage of that imbalance caused by a particular address space. Although the computations involved are beyond the scope of this discussion, the effects of specific coefficients on swap analysis can be studied by considering the maximum possible load balancing values. These values are limited, by definition, to 1/5 of the highest cut-off workload level of all performance objectives.

For example, if the highest cut-off level is 100, the maximum load balancing recommendation value for a load balancer will be (1/5 x 100)=20.

Consider an IPS with WKL=(1,100) and a performance objective with SRV=(500,0). The following graph illustrates the range of variations in the swap recommendation value due to a coefficient of 1 for a load balancer.

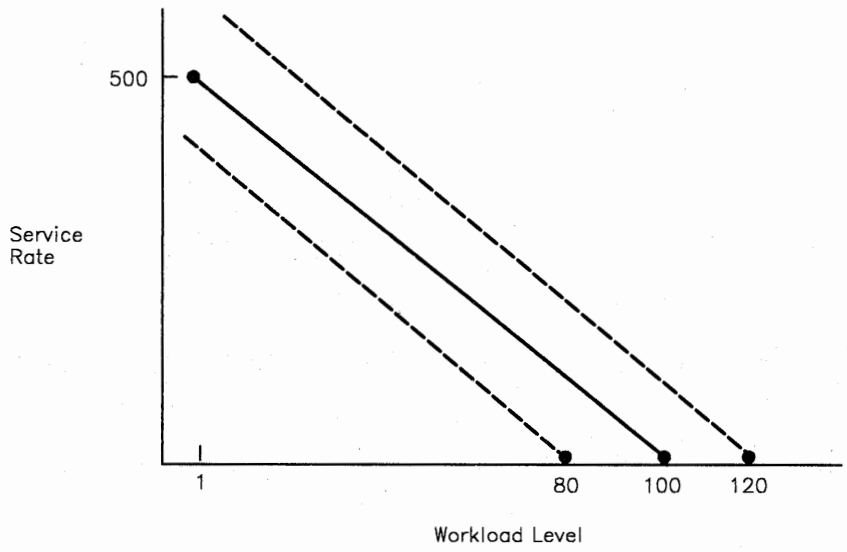


Figure 5-8. The Effect of a Load Balancer Value on Swap Recommendation Values

Figure 5-8 allows the following observations:

- An increase in the load balancer coefficients causes a corresponding decrease in the effect of workload level recommendations on exchange swap decisions.
- Care should be taken in selecting the load balancer coefficients. High load balancer coefficients can cause the load balancing recommendation values to completely override the workload manager recommendation value. This allows the load balancers to continue corrective action when a resource is out of balance without excessive exchange swapping. However, the corrective action might result in address spaces being swapped-out for significant periods of time if the resource is continually out of balance.

  In this case, the concept of equal access to a resource by an address space is abandoned. That is, total throughput is emphasized when using the load balancers, while turnaround time for significant users of a resource in great demand may suffer as a consequence.
- SRM assigns either a positive or negative sign to the load balancing recommendation values, depending on the particular imbalance. For instance, if the CPU is underutilized, SRM will select ready address spaces that are heavy CPU users and add the load balancing value to their workload levels to increase their chances of getting swapped in. If the CPU is overutilized, SRM will select swapped in address spaces that are heavy CPU users and subtract the load balancing value from their workload levels to increase their chances of being swapped out.

Tape Device Selection Options

The procedure for selecting a tape device is described in "Device Allocation," in Section 1. However, the installation can modify the algorithm for the final selection of a tape drive. By using the SELTAPE keyword in the OPT, the installation can choose one of four selection options. These options apply to the eligible devices on the selected logical path. The options are as follows:

- Select the next device number in ascending order (SELTAPE=NEXT). That is, select the device whose device number is higher relative to the last tape unit selected. After the highest device number is reached, selection begins again with the lowest device number. This method is the default.
- Select the first device on the list (SELTAPE=FIRST).
- Select devices at random from the list (SELTAPE=RANDOM).
- Select the device on the list with the lowest device number (SELTAPE=LOWEST). This algorithm tends to use the same device repeatedly.

  See the IEAOPTxx parameter under SRM Parameters in Section 5 of this chapter.

Special Options

The following additional options can also be specified in the OPT.

Transaction Definition for CLISTs: An installation can specify whether or not the individual commands in a TSO CLIST are treated as separate TSO commands for the purpose of transaction control if TSO Extensions is installed. Specifying CNTCLIST=YES causes a new transaction to be started for each command in the CLIST. A possible exposure of specifying CNTCLIST=YES is that long CLISTs composed of trivial and intermediate commands might monopolize a domain's MPL slots and cause interactive terminal users to be delayed. Specifying CNTCLIST=NO (the default) causes the entire CLIST to constitute a single transaction.

In either case the individual commands of a CLIST are not subject to an RTO delay or assigned report or control performance groups through the installation control specification.

Counting Non-swappables in Domain MPLs: The installation can choose to include non-swappable address spaces in the calculation of the current MPL values. If CNTNSW=YES is specified, non-swappable address spaces are included in the sum of the ready users for each domain. Including non-swappable address spaces in the calculation of the MPL value has a stabilizing effect on a system where some of the users are periodically non-swappable for short periods of time. However, in cases where an address space remain non-dispatchable and non-swappable for a long time (for instance, while waiting for a tape mount), specifying CNTNSW=YES might restrict access of ready work to that domain.

When CNTNSW=YES is specified, the minimum MPL for each domain should allow for the permanently non-swappable address spaces in that domain.

Note: The non-swappable system address spaces are usually in the default domain for started tasks and, as such, must be counted when setting the minimum MPL for the domain.

Directed VIO Activity: VIO data set pages can be directed to a subset of the local paging data sets through *directed VIO*, which allows the installation to direct VIO activity away from selected local paging data sets that will be used only for non-VIO paging. With directed VIO, faster paging devices can be reserved for paging where good response time is important. The NONVIO system parameter, in conjunction with the PAGE system parameter, enables the installation to define those local paging data sets that are not to be used for VIO, leaving the rest available for VIO activity. However, if space is depleted on the paging data sets made available for VIO paging, the non-VIO paging data sets will be used for VIO paging.

The installation uses the DVIO keyword to either activate or deactivate directed VIO.

Note: The NONVIO and PAGE system parameters are in the IEASYSxx parmlib member.

Adjusting Constants Options

Certain OPT parameters make it more convenient for installations with unique resource management requirements to change some SRM constants. The defaults provided are adequate for most installations. A parameter needs to be specified only when its default is found to be unsuitable for a particular system environment. Before making any changes, it is advisable in most cases to study the related segments of code in order to be able to predict the impact of the contemplated change on SRM's evaluations and decisions. The following functions can be modified by parameters in the OPT:

- Enqueue residence control
- SRM invocation interval control
- MPL adjustment control
- Logical swapping controls
- CPU management control
- I/O load balancing control
- Storage load balancing control
- Pageable storage control
- Expanded storage control
- Selective enablement for I/O

Enqueue residence control: This parameter, specified by the ERV keyword, defines the amount of CPU service that the address space is allowed to receive before it is considered for a workload recommendation swap out. The parameter applies to all swapped-in address spaces that are enqueued on a resource needed by another user. For more information see "Enqueue Delay Minimization" in Section 1.

SRM invocation interval control: This parameter, specified by the RMPTTOM keyword, controls the invocation interval for SRM timed algorithms. Increasing this parameter above the default value reduces the overhead caused by SRM algorithms, such as swap analysis and time slicing. However, when these algorithms are invoked at a rate less than the default, the accuracy of the data on which SRM decisions are made, and thus the decisions themselves, may be affected.

MPL adjustment control: The OPT provides eleven keywords to specify upper and lower thresholds for the variables that SRM uses to determine if it should increase, decrease, or leave unchanged the MPL. When one of these variables exceeds its threshold value, SRM regards this change as a signal to adjust the MPL. The following chart gives the internal names for the control variables, their thresholds, and conditions that cause a change.

Control variable and internal name	Thresholds that cause an MPL change: decrease	increase	Keyword in OPT
CPU utilization (RCVCPUA)	>RCCCPUTH	<RCCCPUTL	RCCCPUT
CPU utilization (RCVCPUA)	>RCCCPUPH	<RCCCPUPL	RCCCPUP[1]
Page delay time[5] (RCVMSPP)	>RCCMSPTH	<RCCMSPTL	RCCMSPT[1]
Page delay time[5] (RCVMSPP)	>RCCPDLTH	<RCCPDLTL	RCCPDLT[2]
Page fault rate (RCVPTR)	>RCCPTRTH	<RCCPTRTL	RCCPTRT[2]
Demand paging rate (RCVDPR)	>RCCDPRTH	<RCCDPRTL	PAGERT1[1 3 4] PAGERT2
ASM queue length (RCVASMQA)	>RCCASMTH	<RCCASMTL	RCCASMT[2]
UIC (RCVUICA)	<RCCUICTL	>RCCUICTH	RCCUICT
Percentage of online storage fixed or allocated for page-in or page-out (RCVFXIOP)	>RCCFXTTH	<RCCFXTTL	RCCFXTT
Percentage of storage that is fixed within the first 16 megabytes (RCVMFXA)	>RCCFXETH	<RCCFXETL	RCCFXET

[1]The RCCCPUP thresholds are used in conjunction with the RCCMSPT (page delay time) and RCCDPRT (demand paging rate) thresholds when considering an MPL change.

[2]The default thresholds for these keywords cause the corresponding control variables to have no effect on MPL adjustment.

[3]Page fault rate is reported in the RMF summary report as demand paging.

[4]PAGERT1 for a single central processor; PAGERT2 for a multiprocessor.

[5]SRM calculates the page delay time using the paging rate to page data sets and the number of queued paging requests to those page data sets.

An installation with unique resource management requirements might wish to change these variables. Before making such a change it is advisable, in most cases, to study the related segments of code or refer to the method-of-operation diagram for IRARMRM2 in the SRM section of *System Logic Library*, to predict the impact of the change on SRM.

Logical swapping control: Five keywords are provided in the IEAOPTxx parmlib member to influence logical swapping:

1. LSCTMTE specifies the upper and lower values for the think time limit.
2. LSCTUCT specifies the upper and lower UIC threshold values.
3. LSCTAFQ specifies the upper and lower available frame count threshold values.
4. LSCTFTT specifies the upper and lower percentages of online storage that is fixed or allocated for page-in or page-out.

5. LSCTFET specifies the upper and lower percentages of storage that is fixed within the first 16 megabytes.

SRM uses keywords 2, 3, 4, and 5 to determine if the think time limit should be increased or decreased. The relationship between these keywords and their internal values is:

- For the think time limit:

Lower value	%oThink time limit	%oUpper value	Keyword
LSCTMTEL	LSCTMTES	LSCTMTEH	LSCTMTE

- For think time change:

Control variable and internal name	Think time change decrease	increase	Keyword in OPT
UIC (RCVUICA)	<LSCTUCTL	>LSCTUCTH	LSCTUCT
Available frame count (RCVAFQA)	<LSCTAFQL	>LSCTAFQH	LSCTAFQ
Percentage of online storage fixed or allocated for page-in or page-out (RCVFXIOP)	>LSCTFTTH	<LSCTFTTL	LSCTFTT
Percentage of storage that is fixed within the first 16 megabytes (RCVMFXA)	>LSCTFETH	<LSCTFETL	LSCTFET

SRM changes the think time limit as follows:

```
IF (RCVAFQA > LSCTAFQH or RCVUICA > LSCTUCTH) and
    (RCVFXIOP < LSCTFTTL and RCVMFXA < LSCTFETL)
  THEN the think time is increased.
ELSE
  IF (RCVAFQA < LSCTAFQL or RCVUICA < LSCTUCTL or
      RCVFXIOP   LSCTFTTH or RCVMFXA      LSCTFETH)
    THEN the think time is decreased.
```

CPU management control: The keywords provided for CPU management control influence both the CPU load balancing function and the assignment of dispatching priorities in the mean-time-to-wait range.

The CCCUTT keyword specifies the upper and lower thresholds for CPU utilization that are used to determine if the processor is over utilized or under utilized. The CCCUTT keyword affects CPU load-balancing only. The following chart gives the internal names of the control variables and indicates their relation to the condition:

Control variable and internal name	Processor Utilization under	over	Keyword in OPT
processor utilization (CCVLGUTL)	<CCCUTLOT	>CCCUTHIT	CCCUTT

The CCCSIGUR keyword specifies the minimum amount of CPU execution time between I/O waits in determining if a user is a significant user of the processor and should be considered for CPU load balancing. This value is also used in determining the range of user execution times between I/O waits assigned to the different levels of mean-time-to-wait dispatching priorities. This relationship insures that, within a dispatching priority group, the lowest level is assigned to significant CPU users and that the nine remaining priority levels are assigned to users having decreasing mean-time-to-wait values. The CCCSIGUR keyword affects both CPU load-balancing and mean-time-to-wait dispatching.

I/O load balancing control: Four keywords are provided in the OPT to control I/O load balancing. The ICCSIGUP keyword specifies the minimum percentage of user utilization of a logical path in determining a significant user of the logical path. The three remaining keywords specify the threshold values used to determine if a logical path is over utilized or under utilized. There are three types of logical paths; each type has its own keyword. The following chart gives the internal names of the control variables and indicates their relation to the condition:

Control variable and internal name	Logical channel utilization under	over	Keyword in OPT
Logical path utilization (LPBCPUT)	for tape devices ICCLPBLO(1)	ICCLPBHI(1)	ICCLPB(TAPE)
Logical path utilization (LPBCPUT)	for DASD with no dynamic reconnect capability ICCLPBLO(2)	ICCLPBHI(2)	ICCLPB(NDPSDASD)
Logical path utilization (LPBCPUT)	for DASD with dynamic reconnect capability ICCLPBLO(3)	ICCLPBHI(3)	ICCLPB(DPSDASD)

Storage load balancing control: Eight keywords are provided in the OPT to control storage load balancing. The MCCSBFCF and MCCSBSIG keywords provide a scaling factor for the recommendation value and a minimum threshold respectively, to determine if a user is a significant user of storage. The MCCSBSGP keyword specifies the target ratio of the number of significant to insignificant users of storage. The MCCSBINP and MCCSBDSP keywords specify the percentage factors used to increase or decrease the user frame count threshold for significant users. The MCCSBFTH keyword specifies the maximum percentage of storage that is eligible for fixing. The MCCSBAT, MCCSBST, and MCCSBFTH keywords are used to determine if storage is over or under utilized. The following chart gives the internal names of the control variables and indicates their relation to the condition:

Control variable and internal name	Storage utilization under	over	Keyword in OPT
Available frame queue (MCVSBFQA)	>MCCSBATH	<MCCSBATL	MCCSBAT
UIC (MCVSBLTS)	>MCCSBSTH	<MCCSBSTL	MCCSBST
Percentage of storage eligible for fixing that is actually fixed (MCVSBLTF)	<MCCSBFTH	---------	MCCSBFTH

Storage is over utilized if:

```
        MCVSBFQA  <  MCCSBATL
and     MCVSBLTS  <  MCCSBSTL
```

Storage is under utilized if:

```
        (MCVSBFQA  >  MCCSBATH or MCVSBLTS > MCCSBSTH)
and     MCVSBLTF  <  MCCSBFTH
```

The amount by which the threshold value is exceeded directly affects the magnitude of the recommendation value.

Pageable storage control: Two keywords are provided in the OPT to signal a shortage of pageable storage. Keyword MCCFXTPR specifies the percentage of storage that is fixed or allocated for page-in or page-out. Keyword MCCFXEPR specifies the percentage of storage, within the first 16 megabytes, that needs to be fixed before SRM detects a shortage.

Control variable and internal name	Shortage of pageable storage exists	Keyword in OPT
Percentage of storage that is fixed or allocated for page-in or page-out (RCETOTFX +(ASMIORQR-ASMIORQC))[1]	>MCCFXTPR	MCCFXTPR
Percentage of storage that is fixed within the first 16 megabytes (RCEBELFX)[1]	>MCCFXEPR	MCCFXEPR

[1]These variables are actual frame counts rather than percentages. SRM multiplies the MCCFXTPR threshold by the amount of online storage and multiplies the MCCFXEPR threshold by the amount of storage eligible for fixing in order to arrive at the threshold frame counts that it uses to compare against the actual frame counts. If MCCFXEPR x (amount of storage eligible for fixing) is greater than MCCFXTPR x (amount of online storage), then the threshold frame counts that SRM uses to compare against the actual frame counts are set equal.

Expanded storage control: The criteria age is the value used to determine when a page should be sent from real to expanded storage. To modify the criteria age value, eight keywords are provided for optional inclusion in the OPT parameter. The criteria age can be specified for various types of paging groups. Most installations can achieve optimum performance of expanded storage by using the defaults provided for the default criteria table entries.

This is a list of the OPT keywords, and the reasons for page movement for which a criteria age can be specified:

Keyword	Page type
ESCTBDS	Hiperspace (block-addressable) pages
ESCTPOC	Changed page-out pages
ESCTPOU	Unchanged page-out pages
ESCTSTC	Changed stolen pages
ESCTSTU	Unchanged stolen pages
ESCTSWTC	Changed swap-out pages
ESCTSWTU	Unchanged swap-out pages
ESCTSWWS	Working set pages ready for swap-out
ESCTVIO	VIO swap-out pages
ESCTVF	Virtual fetch pages

Except for the ESCTVF keyword (used only for virtual fetch pages), the ESCTBDS keyword (used only for hiperspace pages), and the ESCTVIO keyword (used only for virtual I/O pages), all of the keywords require that a value be specified to describe the type of page. The types of paging groups and their corresponding values are:

0 Privileged or nonswappable address spaces, or from the common area
1 All others not in type 0 or 2
2 Terminal wait swap, or TSO stolen or paged-out

RSM uses two values to determine when a page should be sent to expanded storage: unreferenced interval count (UIC) and migration age.

- The **system-high unreferenced interval count (UIC)** represents the value for the contention of real storage. The system-high UIC measures how long (in seconds) a page has remained unreferenced in real storage. The system-high UIC is actually an inverse measure of real storage contention. That is, when contention for real storage is high, the system-high UIC is low; the number of seconds between references is low, therefore, the page is being referenced frequently. When contention for real storage is low, the system-high UIC is high; the number of seconds between references is high, therefore, the page is being referenced infrequently.
- The **migration age** represents the value for the contention of expanded storage. Migration age measures how long (in seconds) a page has remained unreferenced in expanded storage. Migration age is actually an inverse measure of expanded storage contention. That is, when contention for expanded storage is high, the migration age is low; when contention is low, the migration age is high.

RSM sends swap out trim and swap out working set pages to expanded storage when the sum of the system-high UIC and the migration age is greater than the criteria age. The exception to this are TSO terminal wait, steal, and page out users' working sets. RSM send these to extended storage when the sum of the system-high UIC and the migration age is greater than the criteria age plus the think time. RSM sends changed and unchanged stolen pages, virtual fetch pages, virtual I/O pages, and page-out requested pages to expanded storage when only the migration age is greater than the criteria age.

The purpose of these calculations is to send only those pages to expanded storage that are likely to be referenced again before being migrated to auxiliary storage. The criteria age values are relatively constant, they can be changed only by changing the OPT keywords. The system-high UIC and the migration age are dynamic and change constantly to reflect the current workload on the system.

Selective enablement for I/O: An installation can use the CPENABLE keyword to specify low and high thresholds for the percentage of I/O interruptions to be processed through the test pending interrupt (TPI) instruction. SRM uses these thresholds to determine if a change should be made to the number of processors enabled for I/O interruptions. For more information, see "Selective Enablement for I/O" in Section 1.

The following chart gives the internal names of the control variables and indicates their relation to the condition:

Control variable and internal name	Percentage of I/O interruptions under	over	Keyword in OPT
Percentage of I/O interruptions through TPI instruction (ICVTPIP)	<ICCTPILO	>ICCTPIHI	CPENABLE

Section 3: Guidelines and Examples

This section discusses guidelines for:

- Defining installation requirements and objectives
- Selecting values for IPS, ICS, and OPT parameters
- Evaluating and adjusting these values.

In addition, the default IPS is presented and explained.

An installation's individual needs and requirements may lead to ultimate specifications that differ greatly from the values presented here. The guidelines should, nevertheless, prove useful as a starting point.

Defining Installation Requirements

Before specifying any parameters to SRM, an installation must define response and throughput requirements for its various classification of work.

Examples of specific questions that should be answered are listed below. The applicability of these questions will, of course, vary from installation to installation.

Subsystems

How many subsystems will be active at any one time and what are they?

For IMS or VSPC, how many active regions will there be?

Will the subsystem address space(s) be nonswappable?

What is the desired response time and how will it be measured?

Batch

What is the required batch throughput or turnaround for various job classes?

How much service do batch jobs require, and what service rate is needed to meet the turnaround requirement?

An RMF workload report or reduction of SMF data in type 5 or type 30 records will provide the average service consumed by jobs of different classes. Based on service definition coefficients of CPU = 10.0,IOC = 5.0,MSO = 3.0,SRB = 10.0; the following approximations can be made:

Short jobs use 30,000 service units or less.
Long jobs use 30,000 units or more.

What is the average number of ready jobs?

Most likely, this is the number of active initiators. A few extra initiators may be started to decrease turnaround times.

TSO

What is the number of terminals?

What is the average number of ready users?

As a guideline for installations new to TSO, assume that an installation doing program development on 3270 terminals will have two ready users for every ten users logged on. This average will vary, depending on the type of terminal and on the type of TSO session (data entry, problem solving, program development).

What is the required response time and expected transaction rate for different categories of TSO transactions at different times, such as peak hours?

What is the expected response time for short transactions?

How will this response time be measured?

Should response time be different for select groups of TSO users?

How should semi-trivial and non-trivial transactions be treated?

How are they defined?

An installation can use RMF workload reports or SMF data in type 34 and 35 records available to help define trivial and non-trivial TSO work. Based on service definition coefficients of CPU=10.0,IOC=5.0,MSO=3.0,SRB=10.0; the following approximations can be made:

Short TSO commands use 200 service units or less.
Medium length commands use between 200 and 1000 service units.
Long TSO commands use 1000 service units or more.

What is the required service rate for TSO users?

If 2-second response time (as reported by RMF) is required for very short TSO commands (100 service units), the required service rate for such a transaction is 100/2 or 50 service units per second. Service rates for other types of transactions should be computed also.

General

What is the importance level of TSO, batch, IMS, and special batch classes in relation to one another?

Which may be delayed or "tuned down" to satisfy other requirements?

In other words, which response requirements are fixed and which are variable?

What percentage of system resources should each group receive?

How critical are CPU, real storage, and I/O resources?

If a user makes heavy demands on a resource, the installation will have to decide whether the user's response time requirement should be waived in favor of a more balanced system load.

Once these questions have been answered, and the installation is somewhat confident that its requirements can be met by the hardware, the installation is ready to begin writing an initial IPS, OPT, and installation control specification.

Preparing an Initial Installation Control Specification, IPS, and OPT

This section presents guidelines for selecting installation control specification, IPS, and OPT parameter values. Where applicable, values are given which may be used as initial values when no previous performance data is available.

There are several approaches to preparing an initial installation control specification, IPS, and OPT.

- Use the default IPS and OPT, and no installation control specification.

  It is suggested that installations initially use the default IPS (IEAIPS00) provided. For more information, see "Default IPS" in this section.

- Continue with a current level IPS, OPT, and installation control specification.

  Current levels of the IPS and OPT, although compatible with MVS/ESA, might need updating to obtain similar levels of service.

- Create a new installation control specification, IPS, and OPT (or modify the default IPS).

Work Sheets: If your installation decides to modify the default IPS or create a new IPS and an installation control specification, the following worksheets might be of value while reading this section. They have been included as an aid for recording parameters and values as they are defined. After concluding this section, the completed work sheets should enable an installation to write an IPS, using correct syntax, with a minimum amount of effort. (In most cases, only a subset of the available entries will be needed). Worksheet #5 is an aid for assigning performance group numbers to the various types of work in an installation. From this worksheet, an installation can code a correct installation control specification.

Note: Syntax errors in the installation control specification, IPS, and OPT are indicated by a general message written to the console and more detailed messages written to the system log data set. System log error messages are written following a SET IPS, SET ICS, or SET OPT command, but not during an IPL.

Information.*on

Worksheet Number 1 – Service Definition Coefficients and Domain Information

CPU _____

IOC _____

MSO _____

SRB _____

Workload Levels _____

Highest Cut–Off Workload Level _____

TARGET CONTROL SPECIFICATION

Service Rate

Contention Index

Domain Definitions				
Domain Number	Minimum MPL	Maximum MPL	Weighting Factor or Target Control Keyword and Value	Descriptions

Worksheet Number 2 – I/O and Dispatching Priorities

APGRNG _____ TUNIT _____ TSPTRN _____

	Priority		Associated User Types	
			Dispatching Priority	I/O Priority[1]
	Above APG		Master	
	Above APG			
	Above APG			
Set 9	Fixed	F94		
		F93		
		F92		
		F91		
		F90		
		F9		
	MTW	M9		
		⊙		
		⊙		
		⊙		
Set 3	Fixed	F34		
		F33		
		F32		
		F31		
		F30		
		F3		
	MTW	M3		
Set 2	Fixed	F24		
		F23		
		F22		
		F21		
		F20		
		F2		
	MTW	M2		
Set 1	Fixed	F14		
		F13		
		F12		
		F11		
		F10		
		F1		
	MTW	M1		
Set 0	Fixed	F04		
		F03		
		F02		
		F01		
		F00		
		F0		
	MTW	M0		
	Below APG			

MTW – Mean-time-to-wait

[1] If different from dispatching priority.

Worksheet Number 3 – Execution Characteristics Definition

Group	Period	Duration	ISV Value	Objective Number	Comments

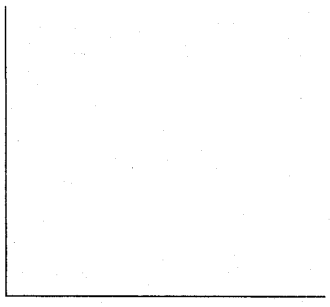
Worksheet Number 4 – Objective Control Per Domain

Workload Levels _____

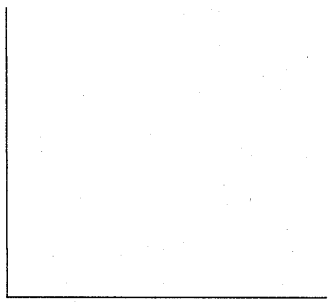
Maximum Absorption Rate _____

Highest Cut – Off Workload Level _____

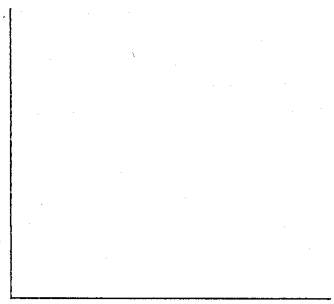
Domain Number 1



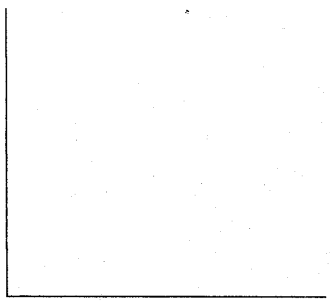
Domain Number 2



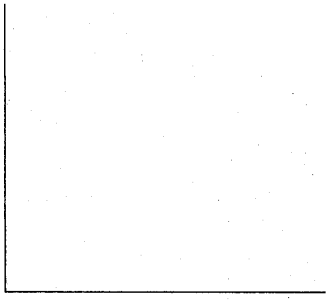
Domain Number 3



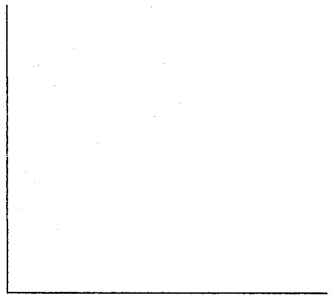
Domain Number 4



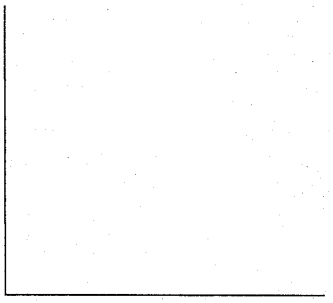
Domain Number 5



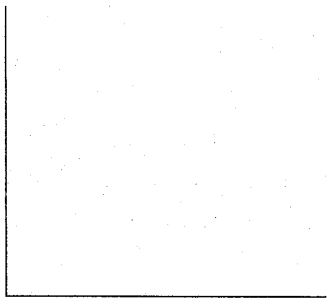
Domain Number 6



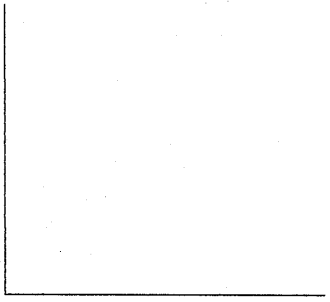
Domain Number 7



Domain Number 8



Domain Number 9



Worksheet Number 5 – Performance Group Number (PGN) Assignment

N/A indicates that assigning a PGN is not meaningful

Subsystem	Transaction Class/Userid/Name	Control PGN[1]	Optional Control PGNs	Report PGN
TSO	TRXNAME=	N/A	N/A	
	USERID=			
	ACCTINFO=			
STC	TRXNAME=			
Job entry subsystem	TRXNAME=			
	USERID=			
	TRXCLASS=			
	ACCTINFO=			
Any Other subsystems	TRXNAME=	N/A	N/A	
	USERID	N/A	N/A	
	TRXCLASS=	N/A	N/A	
	ACCTINFO=	N/A	N/A	

[1] Default control PGN:
- PGN2 for the TSO subsystem
- PGN1 for all other subsystems

Worksheet Number 6 – Performance Group/Period Definition

Performance Group Number		Domain Number	Objective	Dispatching Priority			I/O Priority	Interval Service Value	Duration	RTB	UNT
				DP	TSDP	TSGRP					
Defaults →		See Note 1	OBJ–1	M0	None	Domain Number	IOP–DP or –TSDP	ISV–100K	None	0	S
1 Batch Default	Period 1										S
	2										S
	3										S
	4										S
	5										S
	6										S
	7										S
	8										S
2 TSO Default	1										S
	2										S
	3										S
	4										S
	5										S
	6										S
	7										S
	8										S
	1										S
	2										S
	3										S
	4										S
	5										S
	6										S
	7										S
	8										S
	1										S
	2										S
	3										S
	4										S
	5										S
	6										S
	7										S
	8										S

Note 1: The default is the domain number of the previous period. If no previous period with DMN is specified, the default is DMN=1.

Selecting Service Definition Coefficients (Worksheet Number 1)

The purpose of the service definition coefficients (SDCs) is to allow the individual service components to be weighted. For example, there will probably be a greater demand for the resource in least supply. Using coefficients, service provided by such a resource can be given added importance. This is not to imply that the service values are to be used for accounting purposes, since, as will be seen later, the service consumed by a job is not necessarily repeatable.

The coefficients should be high enough to yield a range of service rates sufficiently high for the workload management function to be effective, but they must not be so high that service rates become excessively large. This also results in ineffective workload management control.

Changes to the coefficients may require changes to other parameters in the IPS that are dependent on service value specifications (for example, interval service value and duration).

An increase in a service definition coefficient will numerically raise the system service capacity and the service rate of users, though not, of course, affecting the system's physical capacity for work.

For example, consider the I/O component of service for the following:

- With IOC = 1.0, a service rate of 100 represents 100 I/O requests/second.
- With IOC = 2.0, a service rate of 100 represents 50 I/O requests/second.

Three alternatives exist for setting the SDCs:

1. Use the values in the default IPS (IEAIPS00). These should generate reasonable service values for the workload management function. The default coefficients are:

   CPU = 10.0
   IOC = 5.0
   MSO = 3.0
   SRB = 10.0

2. Use previous installation SDCs. The installation may wish to include the coefficient for SRB service for several reasons:

   - To obtain information through RMF on SRB usage
   - To monitor SRB usage

3. Define new coefficients. To do this, a more detailed understanding of the individual service components is required. These are discussed separately in the following paragraphs.

CPU and SRB Service: SRM calculates CPU and SRB service based on the task and SRB execution time, the CPU mode, and the number of processors online, which should make the IPS independent of the processor complex.

The following table describes the service consumed per second of execution time by CPU model. The values listed are internal SRM constants. The "total system absorption rate" reported by RMF will not equal the values listed here because these do not include certain types of system processing.

Processor Model	Service Units Per Second of Task or SRB Execution Time	Seconds of Task or SRB Execution Time Per Service Unit
3090 Model 120E	365.22	0.002738
3090 Model 120S	388.89	0.002571
3090 Model 150E	471.91	0.002119
3090 Model 150S	617.65	0.001619
3090 Model 170S	763.64	0.001310
3090 Model 180E	823.53	0.001214
3090 Model 180S	1050.00	0.000952
3090 Model 200E	765.88	0.001306
3090 Model 200S	997.50	0.001033
3090 Model 280E partitioned	823.53	0.001214
3090 Model 280S partitioned	1050.00	0.000952
3090 Model 280E single image	765.88	0.001306
3090 Model 280S single image	997.50	0.001003
3090 Model 300E	724.71	0.001380
3090 Model 300S	966.00	0.001035
3090 Model 400E partitioned	765.88	0.001306
3090 Model 400S partitioned 2 way	997.50	0.001003
3090 Model 400E single image	700.00	0.001429
3090 Model 400S single image	924.00	0.001082
3090 Model 500E partitioned 2 way	765.88	0.001306
3090 Model 500S partitioned 2 way	997.50	0.001003
3090 Model 500E partitioned 3 way	724.71	0.001380
3090 Model 500S partitioned 3 way	966.00	0.001035
3090 Model 500E single image	658.82	0.001518
3090 Model 500S single image	892.50	0.001120
3090 Model 600E partitioned	724.71	0.001380
3090 Model 600S partitioned 3 way	966.00	0.001035
3090 Model 600E single image	625.88	0.001598
3090 Model 600S single image	850.50	0.001176
4381 Model 91E	308.82	0.003238
4381 Model 92E	262.50	0.003810

For installations with no prior service data, the task time reported in type 4, 5, 30, 34, and 35 SMF records could be converted to service units using the above table.

CPU time may not be identical for different runs of the same job step. One or more of the following factors may cause small variations in CPU time: CPU architecture (such as storage buffering), cycle stealing with integrated channels, and the amount of queue searching (see the publication *SPL: System Management Facilities*. An installation may find that SRB time is approximately 1/10 of task time.

I/O Service: SRM calculates I/O service using either I/O block (EXCP) counts or device connect time (DCTI), as specified on the IOSRVC keyword in the IEAIPSxx parmlib member. The calculations are (1) I/O service units equal I/O block count, or (2) I/O service units equal DCTI (in seconds) divided by 8.3 milliseconds. If DCTI is used to calculate I/O service, operations to VIO data sets and to devices that the channel measurement facility does not time are not included in the I/O service total.

When an address space executes in cross memory mode (that is, during either secondary addressing mode or a cross memory call), EXCP counts or the DCTI are not included in the I/O service total. This I/O service is *not* counted for the address

space that is the target of the cross memory reference. For a description of SMF EXCP counts, see *SPL: System Management Facilities.*

Main Storage Service: A program uses one storage service unit when it holds 50 pages (200K) for one CPU service unit. The amount of storage service available can be determined by calculating the number of pages available for problem program use (from RMF paging reports) and applying the following formula:

$$\text{storage service units} = \frac{\text{\# pages x \# CPU service units}}{50}$$

This formula has been simplified to allow the presentation of the following example. In the actual calculation, SRM utilizes a continually updated variable called *page seconds*. This variable is the product of task execution time and the number of frames allocated to an address space, and it is updated periodically. The page seconds accumulated by a transaction are reported in SMF type 4, 30, and 34 records.

The main storage component of service can affect the repeatability of service required by a transaction, because the total storage service available varies with the number of address spaces in real storage at any one time. Thus, an estimate must be made of how many address spaces are in storage at one time in order to determine the total storage service units available.

For example:

Assume 100 pages and 10 CPU service units are available for problem program usage.

Case 1: 1 address space uses all 100 pages for the entire amount of time (10 CPU service units).

The total amount of main storage service used would be:

$$\frac{100 \text{ x } 10}{50} = 20 \text{ storage service units}$$

Case 2: 2 address spaces share the resources equally. That is, each address space uses 50 pages for 5 CPU service units.

The total amount of main storage service used by each address space would be:

$$\frac{50 \text{ x } 5}{50} = 5 \text{ storage service units}$$

Because each address space used 5 storage service units, the total number of units used is 10. All pages and CPU services were used, and yet the total is less than in Case 1. Thus, the service rate for an address space decreases as the number of executing address spaces increases.

Once it is known how many CPU, I/O, SRB, and storage service units are available, the coefficients can be selected to weight each service component. Since CPU service and SRB service measure a transaction's use of the processor, the two coefficients

(CPU and SRB) could be set equal. If the intent is to equalize the importance of the service components, the coefficients may be determined by using the equations:

CPU x (CPU service units + SRB service units) = IOC x I/O service units = MSO x storage service units

Selecting a reasonable value for one coefficient, such as 10 for CPU, allows calculation of the other three values.

Defining Domains and Their Constraints (Worksheet Number 1)

The following suggestions for defining domains are based on separating an installation's work into four general classifications:

Subsystems
Batch
TSO
Special purpose

Each classification is discussed separately. It should be noted that as execution characteristics change, address spaces may be reclassified. For example, very long (to be defined later) TSO commands may be reclassified as batch work.

Guidelines are presented for selecting appropriate MPL minimum, maximum, and adjustment values for each type of domain. (Note that minimum MPL values should not be set too high. This may cause excessive paging overhead.)

Subsystem Domains: One or more domains can be defined for each subsystem, such as IMS, CICS, VSPC, etc. This is useful because the operator command, "DISPLAY DMN," displays the service rate obtained by all address spaces in a domain including the service rate obtained by non-swappable address spaces. If there is no need to display the service rate online, all subsystems can be combined into one domain. Each subsystem should have separate performance groups specified in order to enable RMF to accurately report the service rate received.

Normally subsystems are nonswappable due to the undesirability of incurring swapping overhead. Other reasons may also exist for making subsystem address space(s) nonswappable. For example, the IMS 1.1.1 resource serialization technique does not cause an address space holding a serialized resource to be swapped in should some other address space required that resource. However, if the correct constraint values are chosen (as discussed later), subsystems may run effectively non-swappable and still be swapped out to free frames when they are idle.

MPL specifications for a domain of swappable subsystem address spaces should allow instant swap in of these address spaces when they become ready. Therefore, the minimum MPL should be greater than or equal to the number of address spaces associated with the domain. The maximum MPL value is not meaningful in this case. Since the MPL adjustment function is not utilized for a subsystem domain with these recommended MPL values, the weight or FWKL value of such a domain should be set to 1.

MPL specifications are not meaningful for domains having only non-swappable address spaces. If you do specify a minimum MPL, use a value of at least 1, which allows termination of a non-swappable address space. For domains having both swappable and non-swappable address spaces, the latter can be ignored in setting the MPL specification unless the OPT specifies CNTNSW=YES. Specifying

CNTNSW = YES causes non-swappable address spaces to be counted in the domain's current MPL.

Batch and TSO Domains: There are crucial differences between the execution characteristics of batch and TSO work, such as the amount of service required to complete processing and the arrival rate of work. Batch jobs usually require considerably more service than most TSO commands and could therefore overload the system if the number of such jobs allowed in real storage were not controlled. Such limits, however, are not desirable for most TSO work. Since this control is achieved via the domain constraints, the two different requirements cannot be satisfied by one domain. Therefore, separate domains should be created for batch and TSO work.

In most cases, one domain should be sufficient for batch work. Performance objectives can be used to achieve the desired degree of control over individual jobs.

If all TSO commands require uniform service, one domain will probably suffice to provide good response time. However, if the TSO work consists of commands with diverse service needs, that is, both short and long-running commands, additional domains may have to be defined in order to satisfy these different requirements.

As discussed in Section 2, the maxMPL value is used to limit the number of address spaces allowed in real storage, while the purpose of the minMPL is to guarantee access to real storage for a fixed minimum number of address spaces.

The minMPL value can be used to implement fixed response requirements. If fast response, for example, is a fixed requirement for short TSO commands, the minMPL value can be set to 1 or 2 for every 10 logged on users. For instance, if 40 users are logged on, a minMPL value of 4 would be a good starting point.

For batch work, if there are no fixed throughput requirements, SRM should be allowed to determine the target MPL in a range of 0-999. If, however, a specific number of jobs must be processed during a particular period of time, the minMPL should be raised slightly above zero (1 to 3) to guarantee access to real storage for some minimum number of batch jobs.

For long-running, batch-like TSO commands, SRM should be allowed to determine the target MPL in a range of 0-999.

MPL Target Control

Domain weights and the workload levels derived from the target control objectives have meaning only for domains whose target MPL is adjustable, and only when there is more than one such domain. Domains with fixed target MPLs should be given a fixed workload level of 1 (FWKL = 1) or, if weights are being used, such domains should be given a weight of 1.

Target Control Keywords: Workload levels assigned through the use of target control keywords are, in effect, priorities called contention indexes that vary as relative service rates to domains vary.

Installations that want to control service rates to domains may do so with the target control keywords provided that the following conditions are met:

- The system has sufficient capacity to provide the desired rates.
- The minMPL and maxMPL values permit MPL adjustment to the desired levels.
- There is ready work in the domain.

As an example, a domain could be created for short (interactive) TSO commands specifying a performance objective (using AOBJ=) that requests a constant service rate:

```
WKL=(1,100)
OBJ=1,SRV=(50,50)
DMN=1,AOBJ=1
```

Figure 5-9 shows the contention index and its relationship to service rate. The contention index for the domain is set low or high, depending on whether the domain is or is not receiving the requested service rate (50 service units/second).

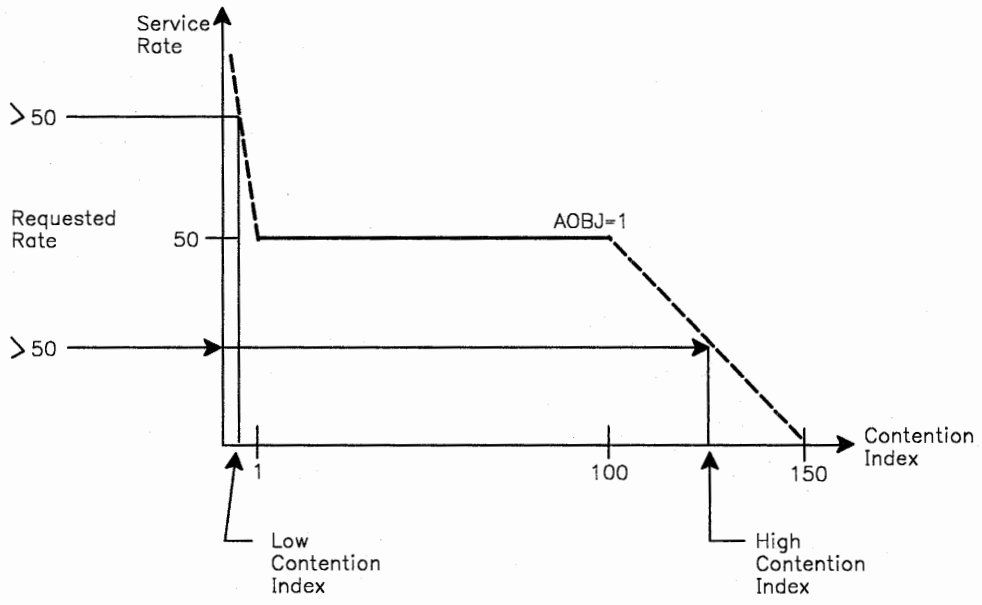


Figure 5-9. Specifying a Contention Index Using AOBJ

Note: If the service rate per ready user equals the requested service rate, the contention index is determined from the rightmost point on the objective. In this example, the contention index is 100 for a service rate of 50.

If an installation defines two domains and wants to provide service to them on a percentage basis, it may request a constant service rate ratio for those domains. For example, a long TSO domain might be given a service rate twice that of a batch domain by using the DOBJ keyword, as shown in Figure 5-10. The contention indexes of the domains will be equal when the service rate of the TSO domain is twice that of the batch domain. This same principal can be applied in requesting a constant service rate ratio for batch classes.

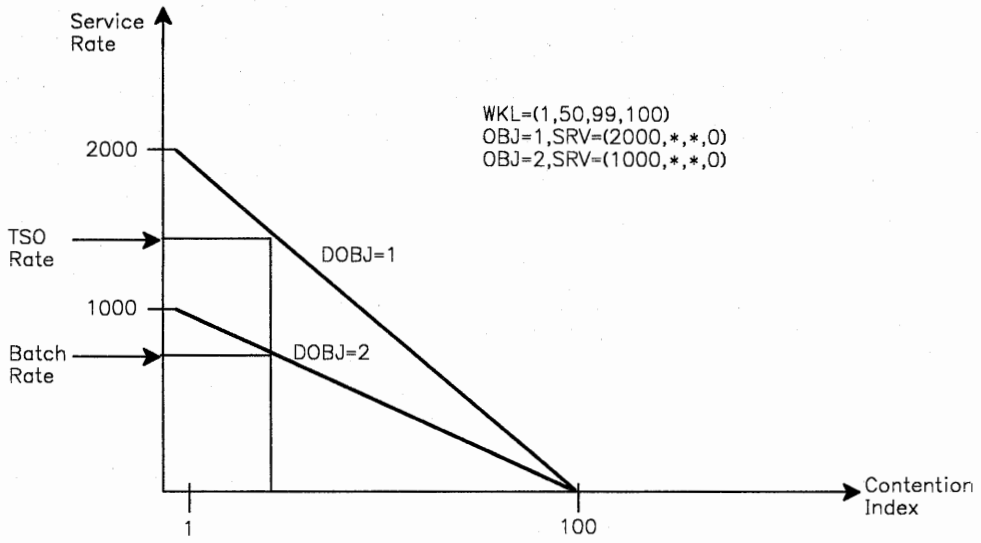


Figure 5-10. Specifying a Contention Index Using DOBJ

An installation may choose to assign a fixed workload level (contention index) to a domain. The example illustrated in Figure 5-11 defines the service rate per ready user as 200 service units/second and a batch domain is defined to receive the remaining service within the constraints of system capacity, MPL limits, and availability of ready work. If the service rate per ready user in the TSO domain (domain 1) decreases to 200 or below (as for instance when the average number of ready users suddenly increases), its contention index becomes 100 or greater and its target MPL will be raised even if this requires the batch domain's target MPL to be lowered. As the service rate per ready user in the TSO domain increases to more than 200, its contention index becomes less than 1, so the target MPL of the batch (domain 2) may then be raised. Thus the service rate of the TSO domain tends to stabilize around 200 per ready user and the batch domain receives the remainder.

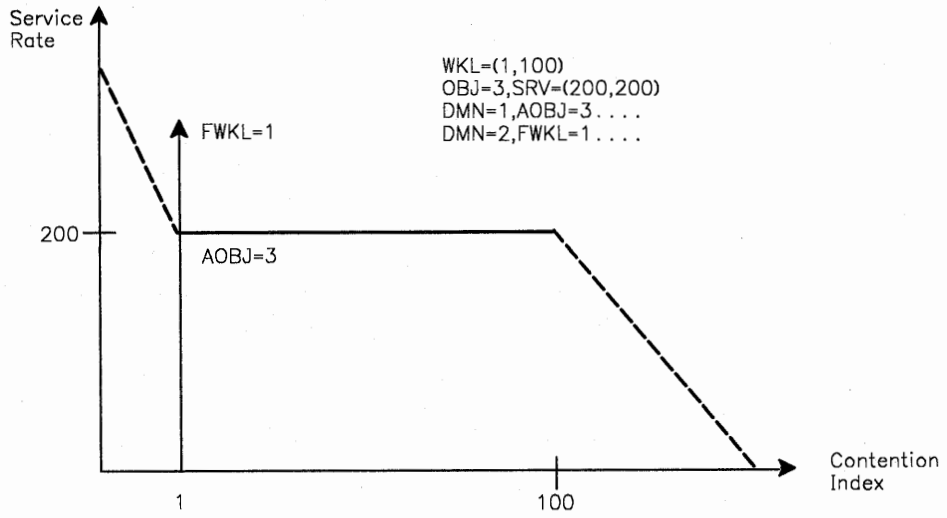


Figure 5-11. Specifying Contention Indexes Using AOBJ and FWKL

Domain Weight: Domain weights have the effect of creating an ordered priority list according to which SRM decides which domain's target MPL should be raised or lowered first. The differences between weight values should be sufficiently large to ensure that the computed contention indexes actually reflect the installation's intentions. These indexes may be calculated beforehand by using representative values, as suggested by the following method:

For batch, assume that the average number of ready users is equal to the number of initiators and that the target MPL is 1.

For TSO, assume that the average number of ready users is equal to the target MPL, resulting in a contention index equal to the weight.

For example, assuming there are 8 initiators, the contention indexes would be:

TSO contention index = TSO weight
batch contention index = 8 x batch weight

Therefore, if the TSO domain is to be favored over the batch domain, the weight for the TSO domain must be more than 8 times the batch domain weight.

Notice that a domain's weight or a domain target control specification has no meaning when the minMPL is sufficiently high to eliminate MPL adjustment.

Note: The use of weight values is not permitted in the same IPS with target control keywords.

Special Purpose Domains: A separate domain may be created to prevent ready work from executing. For example, the system operator may halt the execution of a job via the RESET command by associating it with a domain having a fixed target MPL of zero, that is, minMPL=maxMPL=0.

Selecting I/O and Dispatching Priorities (Worksheet Number 2)

The following recommendations are made for the selection of dispatching priorities:

- Include as many users as possible within the APG. This allows strict control over the dispatching of work. In addition, by including as many users as possible in the APG, the information to control their dispatching is contained in one place as opposed to being spread over several proclib members.
- The relationship (being higher or lower) of dispatching priorities is more important than the actual dispatching priority. Therefore, construct a list of all possible work ordered according to their relative importance before selecting dispatching priorities.
- Address spaces with no fixed requirements, possibly long TSO transactions and long batch jobs, should be assigned to a mean-time-to-wait group of the APG to aid throughput.
- Address spaces that are likely to be CPU-bound should be assigned to the lowes mean-time-to-wait group. The mean-time-to-wait algorithm allows SRM to dynamically adjust the dispatching priority.
- Do not eliminate the benefit of the mean-time-to-wait algorithm. That is, whatever work you are assigning to a mean-time-to-wait group should be assigned to the same group if possible.
- The IMS control region should be assigned a high dispatching priority, possibly lower than that of TCAM or VTAM, but higher than the dispatching priorities of initiators and short TSO transactions.

- The PVLDP keyword should be used to assign a priority to initiators and privileged programs when they have a priority within the APG. The default PVLDP value is the lowest mean-time-to-wait group. The master scheduler's address space and the global resources serialization address space are always assigned the highest priority (X'FF') regardless of the PVLDP value.
- Use the time slice function to prevent one subsystem from dominating another subsystem or group of work. Estimate the percentage of time that the subsystem requires a preferred dispatching priority and use the time slice pattern to meet that percentage. For simplicity, assign the same number to the time slice group as was assigned to the domain.
- Note again that for nonswappable address spaces only the dispatching control specified in the first period of the associated performance group is applicable.
- If the I/O priority should be higher or lower than the dispatching priority, then assign an I/O priority.

Defining Durations (Worksheet Number 3)

Durations may be used to subdivide a major class of work into subclasses. For example, transactions may be subdivided into short, medium, and long. Examples of possible subdivisions by duration are shown in the following table. The values are those used in the default IPS (IEAIPS00).

Notes:

1. Duration may be specified in real time units. However, this practice is not recommended except in special cases, which will be discussed later.

Class	Subclass	Duration (Service Units)
Batch	Short	< 30000
	Long	> 30000
TSO	Short	< 200
	Medium	200 - 1000
	Long	> 1000

   A subclass, in these examples, corresponds to a performance period, with the duration value defining the length of the period. The service units used by TSO sessions and batch jobs are reported in SMF type 4, 5, 30, 34, and 35 records. An installation may use this data to define appropriate subclasses. For batch jobs, duration values may be selected to reflect initiator classes.

2. Performance groups that are used exclusively for nonswappable applications should have no more than one performance group period. Period switching is not done for nonswappable address space.

Selecting ISV Values (Worksheet Number 3)

As discussed in Section 2, ISV values are used to control the frequency of swapping. An increase in ISV values reduces the number of exchange swaps. The effectiveness of ISV control over swapping is reduced if multiple performance objectives are used in a domain and these have cut-off levels that differ by more than one.

When duration specifications exist, ISV values for the respective periods will depend directly on the duration values. For example, if short TSO transactions have a duration of 200 service units and exchange swaps are to be prevented during this interval, the ISV value must be at least 200 service units. If an ISV value of 40,000

were associated with short batch transactions (see table), no swaps would occur during that period, since the transaction would switch periods after 30,000 service units. If the ISV value were set to 15,000, one exchange swap would probably occur after 15,000 service units, provided, of course, that other address spaces of the domain are ready to be swapped in.

The accumulation of service is not reset when an address space is associated with a new performance group period. If exchange swaps are to be avoided in the succeeding period, the ISV should include the duration of the new and all previous periods.

For example, if the first period TSO transactions have a duration of 200 service units (as in the previous example), and the second period transactions have a duration of 800 service units, then, the ISV for the second period should be at least 1,000 (200 plus 800) to avoid exchange swaps during the second period.

If the third period is the final performance group period and specifies an ISV of 10,000, then a TSO transaction is allowed to consume 9000 (10,000 minus 1000) service units after switching to period three while maintaining the same cutoff workload level.

Each time an address space is swapped in, service accumulation is reset to zero. For example, assume that a known TSO transaction consumes 18,000 service units and was swapped-out when it consumed 10,000 service units. Because ISV = 10,000 for period three and the transaction requires only 8000 service units, the transaction, when swapped-in, is able to complete within the cutoff workload level (assuming period three is the final period).

ISV values are not meaningful for non-swappable address spaces and for domains whose minMPL is sufficiently high to allow all its ready address spaces to be swapped in at one time.

Selecting Performance Objectives (Worksheet Number 4)

To draw a performance objective, workload level numbers and service rates must be selected. For recommended workload level numbers, refer to "Default IPS" and "IPS Examples."

An appropriate set of service rates should take into consideration the total available system service as described under "Selecting Service Definition Coefficients."

Adjustments may be necessary after analysis of RMF workload reports showing the absorption rates of individual performance objectives. If the absorption rate of an objective approaches the maximum specified service rate, that objective should be redefined with a higher maximum service rate.

The following points should be considered when defining performance objectives to control work within a domain:

- Should the cut-off workload level be equal to that of other objectives in the same domain, or should address spaces be preempted while in their ISV interval? For example, especially important batch jobs may be associated with an objective that allows them to preempt all other jobs in the domain.
- The slope of an objective is not meaningful when all transactions in a domain complete processing within their ISV interval.

- Drawing a steeper slope will cause recommendation values for swapped in address spaces not in their ISV to decrease less rapidly, thereby allowing more service to be accumulated prior to swapping. The steeper the slope of the objective, the less swapping done.
- Objectives should be kept simple.

The following points should be considered when defining performance objectives for use in domain target MPL control:

- Objectives should be described to keep the average service rate within an installation prescribed range (100-150 service units/second).
- When defining objectives for target control keywords, remember that keywords map service rates into domain MPLs. Keep in mind how system services should be allocated when sufficient resources are not available.
  - To provide one domain twice the service rate of another domain, define two objectives (for use with DOBJ), where one objective when mapped, has twice the slope of the other.
  - If a horizontal objective is to be used with AOBJ control, be aware of the cut-off workload level achieved through extrapolation (as discussed in the SRV syntax) when using FWKL values.

For batch work, two objectives may provide the initial desired degree of control, while one objective may be sufficient for TSO work. Suggested variations for both types are described under "IPS Examples." One possible variation of the use of objectives is illustrated in the following figure.

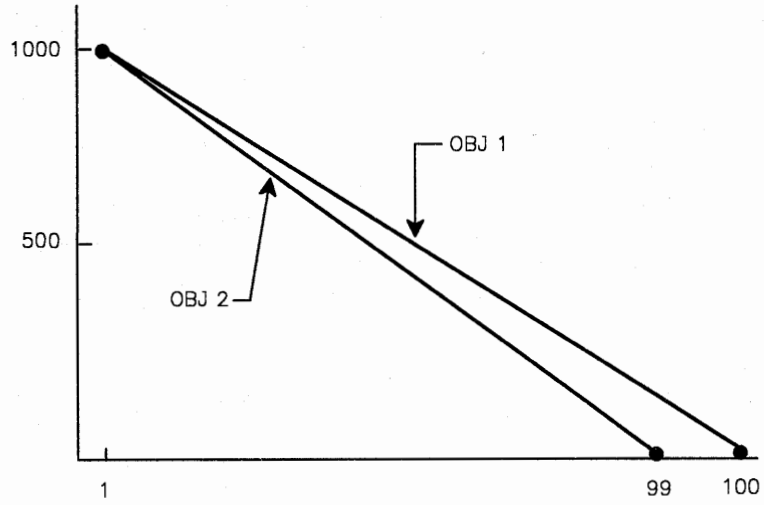


Figure 5-12. A Use of Different Cut-Off Levels

Explanation:

Workload levels must differ by more than 1 to cause an exchange swap. Therefore, address spaces associated with the lower objective will not be preempted by address spaces associated with the higher objective while in their ISV intervals. That is, ISV preemption will not occur even though the cut-off levels of the two objectives are different.

If, however, the MPL adjustment function were to decrease the target MPL of this domain, the lower cut-off level of objective 2 would cause an address space associated with that objective to be swapped out. If the cut-off levels were equal, an address space associated with objective 1 would have the same chance to be selected for swap out.

This combination of objectives, therefore, accomplishes a dual purpose - ISV preemption is avoided, and the intended function of the higher objective is preserved.

Performance Group Number Assignment (Worksheet Number 5)

Worksheet number 5 aids in writing an installation control specification (IEAICSxx parmlib member). The worksheet already includes the system-defined subsystems TSO and STC. To these, add the installation's job entry subsystem and any interactive system that uses the SRM reporting interface. For each subsystem, use the proper keywords (TRXNAME, USERID, TRXCLASS, ACCTINFO) to list the types of work for which a control or report PGNs are to be assigned. Where possible, use sub-string notation to indicate groups of users or transactions. For example, group all users with the same department number together. Another example would be to group all similar batch jobs together.

Control PGN assignment: Reasons for assigning unique control PGNs vary from installation to installation. The following are possible candidates for unique numbers:

- Each subsystem
- Individual subsystem address spaces
- Groups of users
- Departments
- Batch initiator classes
- Monitor programs

Assign control PGNs to the various types of work within the listed subsystems. On the first line, assign a default control PGN for each of the subsystems. When assigning control PGNs, keep in mind the searching order for the installation control specification performance group entries. It is possible for a transaction to match several installation control specification entries, but only one entry is used to assign the control PGN.

Also, remember that if a transaction does not match any entry in the installation control specification, and the subsystem does not specify a default PGN, the transaction is assigned the system default. The system defaults are listed on the bottom of the worksheet.

After assigning the control PGN, list any optional control PGN. An optional PGN is one that the user is allowed to specify through the JCL or LOGON PERFORM parameter. Also, be sure to define control performance groups in the IPS. Worksheet number 6 aids in writing the control performance group definitions.

Report PGN assignment: In order to produce separate RMF workload activity reports for different types of work, assign unique report PGNs. A report PGN can be specified whether or not a control PGN is specified. Report PGNs can be specified for any subsystem defined in the installation control specification. The following are possible candidates for unique report PGNs:

- Each subsystem
- Departments

- All TSO work, including the TCAM address space
- Individual TSO commands
- Groups of commands for an interactive subsystem
- Groups of batch jobs (grouped by class, job name, or userid)

Assign report PGNs where necessary. Remember that, in some cases, the information in the control PGN report is adequate. However, the control PGN might not report on all the transactions because a control PGN collects data for only those transactions that are assigned the PGN as a result of the hierarchical search of the installation control specification.

Note: RPGNs for TSO commands are meaningful only if TSO Extensions is installed or if the interactive subsystem uses the SRM reporting interface.

Performance Groups (Worksheet Number 6)

Once performance groups are assigned, decide if the control over the work in the group should change as the group ages. If so, establish a performance period by assigning a duration value. It is recommended that service units be used as the units for the duration value.

Assign the domains and the dispatching priorities for each period. In those periods where swapping is desirable for throughput control (high priority batch, low priority work, etc.), assign an objective. An objective is not needed for periods describing non-swappable jobs or periods describing a domain where exchange swaps are not desired (TSO trivial and TSO medium). In those periods where swaps are desired to equalize response time, assign an interval service value. For other periods, the default of 100K will eliminate exchange swapping. The response-throughput bias (RTB) should be used only for periods where swap recommendations are to be based on throughput, not response time.

Note: To conserve space, the worksheet omits the storage isolation parameters and the TSO response time parameter. These parameters are used only in exceptional cases.

Selecting Storage Isolation Values

As described in Section 2, storage isolation is used to protect the working set of an application or the common area. Storage isolation can be requested by specifying:

- only the working set size thresholds
- only the page-in rate thresholds
- both the working set size and the page-in rate thresholds

Working Set Size Isolation: This is the simplest form of storage isolation control and can be used for the common area or applications when the working set is well-defined and constant.

To define working set size storage isolation, use the CWSS and PWSS keywords. Specify a minimum working set size that is slightly larger than the critical working set. SRM makes the target working set size equal to the minimum working set and does no additional adjusting of the target. In specifying the maximum working set size, consider that the number of allocated frames exceeding the maximum working set size are preferred for stealing. Therefore, either specify an asterisk (*), which will give you the maximum available storage, or specify a maximum working set size larger than the number of allocated frames. If you want to limit the working set, specify a maximum working set smaller than the number of allocated frames.

When specifying the working set size, remember that it describes common and private area pages in processor storage (either real or extended storage.)

In specifying limits for CWSS, consider that the common working set consists of all CSA, SQA, and PLPA frames even though SQA frames can not be stolen.

Page-in Rate Isolation: This form of storage isolation can be used in cases where the working set is usually constant but can vary with load variations.

To use page-in rate isolation, determine a page-in rate that is acceptable. Specify the page-in rate using the CPGRT keyword for the common area, and either the PPGRT or the PPGRTR keyword for an address space within a performance group period. (You can specify CPGRT once for an IEAIPSxx member; you can specify either PPGRT or PPGRTR once for each performance group period in an IEAIPSxx member.) SRM initially sets the target working set size to zero. Normal stealing occurs until SRM, by periodically adjusting the target working set size, protects enough frames to bring the page-in rate within its thresholds.

The page-in rate isolation control should not be used to protect applications that constantly generate a high page-in rate independently of the number of allocated frames. In this case, the working set size for the application could grow very large and have little effect on the application's performance but a negative effect on total system performance.

Page-in Rate Calculation: SRM's method of calculating the page-in rate for most address spaces depends on whether you specify the PPGRT keyword or the PPGRTR keyword. If you use the PPGRT keyword, SRM calculates the page-in rate based on the number of non-swap, non-VIO page-ins per second of **execution** time. If you use the PPGRTR keyword, SRM's calculation is based on the number of non-swap, non-VIO page-ins per second of **residency** time.

The page-in rate for a cross memory address space is always defined as page-ins per second of **elapsed** residency time because of the operational characteristics of a cross memory address space.

A cross memory address space is defined as one that can be entered via a cross memory program call (PC) instruction from another address space. Examples of cross memory address spaces are the system component address spaces, such as the PC/AUTH address space, the global resource serialization address space, the allocation address space, or any subsystem-defined address space established for cross memory entry. The page-in rate for these address spaces is based on elapsed time because these address spaces are referenced through cross memory functions and thus do not accumulate enough execution time to allow the calculation of a meaningful page-in rate.

The page-in rate for the system common area is also based on *elapsed* time. The rate is defined as the number of CSA and PLPA page-ins per second of elapsed time. Like the cross memory address space page-in rate, the common area page-in rate is based on elapsed time because execution time is not accumulated for the common area. The common area and a cross memory address space are similar in that they both contain data areas and programs that are referenced by multiple address spaces.

Working Set and Page-in Rate Isolation: This combination is the most effective form of storage isolation because the target working set size can vary between the minimum and the maximum working set size based on the actual page-in rate. SRM initially sets the target working set size to the minimum and periodically adjusts the target up or down to keep the page-in rate within its thresholds.

Use the CWSS and the PWSS keywords to specify a minimum working set size smaller than the lowest critical working set size expected for the application or common area. In specifying the maximum working set, consider that the number of allocated frames exceeding the maximum working set size are preferred for stealing. Therefore, either specify an asterisk (*), which will give you the maximum available storage, or specify a maximum working set size larger than the number of allocated frames. If you want to limit the working set, specify a maximum working set size less than the number of allocated frames.

When specifying the working set size, remember that it describes common and private area pages in processor storage (either real or extended storage). Specify a working set size high enough to prevent replacement of too many pages from extended storage, but not so high that it prevents the replacement of enough pages.

In specifying limits for CWSS, consider that the common working set consists of all CSA, SQA, and PLPA frames even though SQA frames can not be stolen.

The page-in rate specification enables SRM to adjust the target working set size as the critical working set varies due to load variations. To specify a page-in rate, use the CPGRT keyword for the common area, and either the PPGRT or the PPGRTR keyword for an address space within a performance group period. (You can specify CPGRT once for an IEAIPSxx member; you can specify either PPGRT or PPGRTR once for each performance group in an IEAIPSxx member.)

Note: The PPGRT keyword is generally not effective for TSO performance group periods 1 and 2. SRM requires a certain amount of history before it can calculate the paging rate for an address space. SRM resets this history for each new transaction, and, because most TSO transactions are so short, SRM never calculates a nonzero paging rate. As a result, the target working set size remains at the minimum specified on the PWSS keyword. PWSS can still be used for TSO address spaces to maintain a minimum working set size across swap-outs, and, with sufficient real storage available, you can allow larger swap-in sets in order to reduce demand page-ins for trivial transactions.

The RTB Parameter and the Resource Factor Coefficients

It is recommended that the RTB be set to zero for swappable address spaces with specific response time requirements (for example, short TSO transactions and high priority batch work).

Address spaces that have no specific response time requirements and often place a significant demand on resources that are sometimes over or under utilized are good candidates for a nonzero RTB specification.

Setting the RTB to 1 allows SRM to monitor all types of resources and dynamically determine which type of resource requires load balancing.

Setting the RTB to any combination of S, C, or I activates the storage, CPU, or I/O load balancing functions, respectively. These RTB values can be used for an address space that places a significant load on a particular resource that is sometimes under

or over utilized. However, using the load balancers effectively does not require the installation to have detailed knowledge about the resources its transactions require.

The installation specifies a particular load balancing function by means of the RTB parameter in the IPS and a weighting factor for the load balancer in the OPT. If a particular load balancer is not required, that is, there is little or no adverse contention for a particular resource, it is advisable to set the weighting factor in the OPT to zero to deactivate the monitoring and save some SRM processing.

As discussed in Section 2, the storage, CPU, and I/O load balancing recommendations values are added to the workload level recommendation. The recommendation values from the load balancers (which can be positive or negative), are limited to one-fifth of the highest workload level specified or implied in the current IPS. Therefore, depending on the weighting factor specified in the OPT, it is possible for the load balancer swap recommendation to completely override the workload level swap recommendation. Large weighting factors imply that throughput is emphasized and exchange swapping is minimized. Small weighting factors imply that throughput is a concern but that an application should be given access to resources even if the application is a significant user of an over utilized resource. This access to resources is possible because the workload level recommendation value increases with time for a swapped-out address space and can override a static load balancer recommendation value.

Default IPS

The default IPS is based on the guidelines just presented. However, not all specific recommendations were used because this IPS must be acceptable for any CPU, storage, and I/O configuration. The philosophy of this default IPS is to provide highest priority (best response) to short TSO commands, then medium TSO commands, then long TSO commands, and to detain batch jobs whenever TSO service needs to be increased.

Figure 5-13 shows the IPS that is available as the default via an IEBCOPY during the installation of MVS/ESA.

```
/*      THIS IPS PREFERS TSO SHORT ABOVE ALL OTHER WORK.  SHORT IS          */
/* DEFINED AS 200 SERVICE UNITS.  LONG TRANSACTIONS IN TSO ARE             */
/* PREFERRED OVER BATCH BY ALLOWING THEM TO RECEIVE TWICE THE              */
/* SERVICE RATE THAT BATCH DOES.  DEFAULTS ARE TAKEN WHERE                 */
/* APPROPRIATE TO MAKE THE MEMBER MORE READABLE.                           */
/*                                                                         */
APGRNG = (0-15)                          /* ALL DISP PRTY IN APG           */
PVLDP = F54                              /* PRIVILEGED USER DPRTY          */
IOQ = PRTY                               /*                                */
CPU = 10.0,IOC = 5.0,MSO = 3.0,SRB = 10.0  /*                              */
/*                                                                         */
/*                                                                         */
/* OBJ1---(DEFAULT) - A STEEP SLOPE                                        */
/*              (USED WHERE EXCHANGE SWAP CONTROL IS NOT REQUIRED          */
/*              TO EQUALIZE RESPONSE TIMES AND THROUGHPUT).                */
/* OBJ2---MODERATE SLOPE                                                   */
/*              (USED TO EVENLY DISTRIBUTE SERVICE TO LONG BATCH           */
/*              AND LONG TSO TRANSACTIONS).                                */
/* OBJ3&4 - ONE SLOPE IS TWICE THE OTHER                                   */
/*              (TO GIVE TSO LONG TWICE THE SERVICE RATE OF BATCH          */
/* OBJ5---A HORIZONTAL LINE                                                */
/*              (USED FOR IMPORTANT WORK TO PREEMPT OTHER BATCH)            */
/* OBJ6---UNFAVORABLE SLOPE                                                */
/*              (USED FOR UNIMPORTANT WORK THAT SHOULD BE DELAYED)         */
/*                                                                         */
/*                                                                         */
WKL = (1,50,99,100)
OBJ = 2,SRV = (2000,*,0)                 /* LONG BATCH AND LONG TSO        */
OBJ = 3,SRV = (2000,*,*,0)               /* DOBJ FOR TSO                   */
OBJ = 4,SRV = (1000,*,*,0)               /* DOBJ FOR BATCH                 */
OBJ = 5,SRV = (2000)                     /* HOT BATCH                      */
OBJ = 6,SRV = (2000,1,0)                 /* LOW PRIORITY BATCH             */
/*                                                                         */
/*                                                                         */
/* ALL DOMAINS HAVE DEFAULTED MINIMUM AND MAXIMUM MPL (1,255)              */
/* BUT EXPLICITLY INDICATE THEIR CONTENTION INDEX ALGORITHM.               */
/*                                                                         */
/*                                                                         */
DMN = 1,DOBJ = 4                         /* BATCH                          */
DMN = 2,FWKL = 128                       /* SHORT AND MEDIUM TSO           */
DMN = 3,DOBJ = 3                         /* LONG TSO                       */
/*                                                                         */
/*                                                                         */
/* DOMAINS PROVIDES ACCESSIBILITY TO MAIN STORAGE.                         */
/* ISV (DEFAULT 100K) ENSURES RESIDENCY IN MAIN STORAGE FOR MOST,          */
/*      BUT ALLOWS EXCHANGE SWAPS FOR LONG TSO TRANSACTIONS.               */
/* DURATIONS ALLOW CONTROL PARAMETERS TO CHANGE AS TRANSACTIONS AGE.       */
/* RESPONSE THROUGHPUT BIAS FAVORS RESPONSE FOR ALL                        */
/* OBJECTIVES ARE USED AS DESCRIBED ABOVE.                                 */
/*                                                                         */
/*                                                                         */
PGN = 1, (DMN = 1,DP = M2,DUR = 30K)     /* BATCH - SHORT                  */
         (DMN = 1,DP = M2,OBJ = 2)       /* -----   - LONG                 */
PGN = 2, (DMN = 2,DP = F34,DUR = 200)    /* TSO     - SHORT                */
         (DMN = u,DP = F32,DUR = 800)    /* -----   - MEDIUM               */
         (DMN = 3,DP = M2,OBJ = 2,ISV = 10K)  /* -----   - LONG            */
PGN = 3, (DMN = 1,DP = F30,OBJ = 5)      /* HOT BATCH                      */
PGN = 4, (DMN = 1,DP = M1,OBJ = 6)       /* LOW PRTY BATCH                 */
```

Figure 5-13. The Default IPS for MVS/ESA

IPS Examples

Although the following examples contain typical values, the examples are not intended as guidelines or recommendations for specific situations.

Example 1

Assume:

- An IMS/Batch system that is running APL/SV.
- Batch class X has a 1 hour turnaround time requirement. All other batch work is class A and has an overnight turnaround requirement. Initiators are started to class XA.

```
CPU=10,0,IOC=8.0,MSO=1.1              /* IPS Example 1                    */
APGRNG=(0-15)                         /* IMS,APL,BATCH ALL IN APG         */
TUNIT=3                               /*                                  */
PVLDP=F50                             /* INITIATOR DISP.  PRIORITY        */
TSPTRN=(*,3,*)                        /* APL at a low dispatching priority */
                                      /*  66% OF TIME                     */
WKL=(1,100)
OBJ=2,SRV=(2000,0)
OBJ=3,SRV=(100,0)
DMN=1,FWKL=2                          /* BATCH-MPL WILL VARY              */
DMN=2,FWKL=1                          /* IMS-NON SWAPPABLE                */
DMN=3,FWKL=1                          /* APL/SV-NON SWAPPABLE             */
PGN=1,(DMN=1,DP=M1,OBJ=3)             /* BATCH EXCEPT CLASS X             */
PGN=2,(DMN=2,DP=F70)                  /* IMS                              */
PGN=3,(DMN=3,DP=F0,TSDP=F6,TSGRP=3) /* APL/SV                           */
PGN=4,(DMN=1,DP=M1,OBJ=2)             /* BATCH CLASS X                    */
```

Notes on Example 1:

Service definition coefficients were chosen to be consistent with previously used values.

Performance groups 2 and 3 describe non-swappable subsystems. As indicated, the only control for these groups is the dispatching priority. Domains are assigned to obtain domain service rate information on the console when the operator specifies a "D DMN" command.

Domain 1 is the only domain that has an MPL that will vary (non-swappable subsystems are not counted in current MPL). Therefore, a simple FWKL adjustment control is sufficient.

Example 2

Assume:

- A CICS/TSO/BATCH system with a major emphasis on CICS.
- CICS uses the byte multiplexer channel 0 very heavily, as does JES with its high speed TP lines.
- TSO activity is light with very few users logged on. Average service units per trivial command, as obtained from prior RMF reports is 110.

```
CPU=10,0,IOC=5.0,MSO=0.0
IOQ=PRTY
APGRNG=(0-15)
PVLDP=F5
WKL=(1,100)
OBJ=2,SRV=(150,150)
OBJ=3,SRV=(1000,0)
DMN=1,FWKL=2
DMN=2,CNSTR=(2,10),AOBJ=2
DMN=3,CNSTR=(1,1)
DMN=4
DMN=5,CNSTR=(0,0)
PGN=1(DMN=1,DP=M1,OBJ=3)                /* BATCH   */
PGN=2(DMN=5,DP=F33,DUR=1,UNT=R)         /* TSO     */
     (DMN=2,DP=F33,DUR=150)
     (DMN=2,DP=F32,DUR=350)
     (DMN=3,DP=M1,OBJ=3,ISV=10K)
PGN=3,(DP=F9)                           /* JES     */
PGN=4,(DMN=4,DP=F9)                     /* CICS    */
PGN=5,(DMN=5,DP=F6)                     /* TCAM    */
```

Notes on Example 2:

- Good CICS response time is dependent on its accessibility to the channel. For this reason, I/O requests are queued by priority and CICS is placed at a fixed dispatching priority equal to JES. The fixed, equal dispatching priority is used so that JES, which is started first, will always be ahead of CICS on the dispatching queue, but their I/O requests will be queued equally, ahead of all other work.
- The DPRTY keyword has been eliminated from the JES proc so that its dispatching priority falls within the APG.
- All TSO commands are delayed access to resources for 1 second (DMN 5 maxMPL=0). This allows the installation to expand TSO usage in the future with a minimum impact on response time.
- Service units for JES are accumulated in domain 1. CICS units in domain 4, and TCAM units in domain 5. Since TCAM is non-swappable, it remains in storage despite the constraint values of domain 5.

Evaluating and Adjusting the IPS and OPT

This section is intended as an aid in interpreting measurement data and adjusting the IEAIPSxx and IEAOPTxx parameter values.

The first part of this section addresses possible causes for a failure to meet fixed response and throughput requirements. The analysis relies on RMF reports, and other measurement data as established by the installation when defining the fixed requirements.

The second part addresses the optimal use of SRM control mechanisms, that is, how to achieve fixed requirements with a minimum of swapping and paging overhead, and how to maximize throughput via job mix control.

Note that the controls offered by SRM are only one aspect of the tuning process. Therefore, adjustment of IPS and/or OPT parameters may not be effective if basic system tuning is needed.

Fixed Requirements Are Not Being Met

Problem 1: Short TSO Response Time is Erratic.

Detection: User feedback or personal experience while using TSO will be the best detection method. Also, RMF reports standard deviation as well as the average response times.

Possible Causes:

1. The dispatching queue is not ordered properly.

   If short TSO transactions are placed on the dispatching queue behind relatively long transactions, long response times can occur. Use the dispatching control function in the IPS to place long TSO transactions at a lower dispatching priority than first period TSO transactions, and ensure that the long transactions are indeed switching to a new period (DUR value for short TSO transactions is not too large).

   Also, examine the use of dispatching priorities higher than or equal to the short TSO transaction priority. There may be dispatching queue interference from other work which has a higher dispatching priority.

2. The number of TSO users waiting to be swapped in is not uniform. Sporadic waiting periods translate into user-perceived erratic response time. RMF tracing reports will indicate the number of users swapped out for the short TSO transaction domain. If this value fluctuates, there are several possible solutions:

   a. Increase the target MPL for the short TSO transaction domain by raising the minimum MPL above the observed average number of address spaces. The minimum MPL may need to be increased several times, even to the mean observed maximum number of ready address spaces. This helps ensure that when a TSO address space becomes ready, it is swapped in. This provides consistent short TSO transaction response, but may also mean that short TSO transactions consume more system resources than is tolerable, introducing, for example, sudden heavy page demands.

   b. Decrease the target MPL for the short TSO transaction domain by lowering the maximum MPL. This should produce a steady number of swapped-out TSO transactions, thus providing a more consistent delay for all transactions and the elimination of erratic response time. It may, of course, also mean an overall increase in response time and a lower transaction rate for all TSO transactions.

c. Use the RTO parameter in the first period of the TSO performance group, specifying the response time for first period TSO transactions. If necessary, the RTO function delays TSO commands to achieve the response time objective. The RTO function provides more consistent TSO response times during system workload fluctuation. If small, the RTO delay might not be noticed at the terminal.

d. The objective specified by the target control keyword AOBJ for the trivial TSO transaction domain limits the domain's MPL. The response time may be erratic if the objective specifies an average service rate that is attained when few address spaces are in real storage and several are swapped out. Change the objective so that a greater service rate is specified, or examine the contention index for each domain to understand why the TSO trivial transaction domain is not chosen for MPL adjustment.

Problem 2: TSO Response Time Is Too Fast.

Detection: The objective criteria established when defining installation requirements is the only means of judging this condition. Both SMF (records 34 and 35) and RMF provide service usage and transaction information.

Possible Causes:

1. TSO domains are too heavily favored.

   As TSO address spaces become ready, they are being allowed instant access to system resources. This may be slowed in several ways:

   a. Use the RTO parameter as described in Problem 1.

   b. Lower the maximum MPL for the short TSO domain to the average number of ready address spaces as indicated by the RMF report.

      This eliminates the slight advantage the domain would have received in the instances where the target MPL had previously exceeded the average number of ready users.

   c. Use RMF trace to examine the contention index for each domain. If the TSO domains are favored heavily, adjust the weighting factor, or the objectives specified by target control keywords, to obtain the appropriate domain priority scheme.

2. TSO has too great a dispatching priority advantage. Possibly, long TSO transactions should use the same range of dispatching priorities as batch work. This will help delay long TSO response time, but will not slow down short TSO response. However, placing any address spaces at an equal or higher priority than short TSO transactions may cause erratic response time, and thus is a fairly drastic solution.

Problem 3: TSO Response Time is Poor.

Detection: Once again, only the criteria established while defining installation requirements can identify this problem. SMF (records 34 and 35) and RMF provide service usage and transaction information.

Possible Causes:

1. The maximum MPL for the short TSO transaction domain is too low, and/or the calculated contention index is too low.

   If there are always more users ready than the maximum MPL will allow into storage, the time each of them spends waiting to be swapped in may be causing the long response time. If the RMF trace report indicates a consistent supply of ready but swapped-out users in the TSO domain, the maximum MPL, and possibly the MPL adjusting parameters (FWKL, AOBJ, DOBJ, and WT), should be changed to allow the TSO target MPL to rise.

2. The minimum MPL is too small.

   Under fluctuating load conditions, the MPL adjusting function may not be able to react quickly enough to high bursts of demand for resources, since it may not raise the target MPL to the maximum of ready users. Instead, the target MPL will remain very stable.

   If this target is too low to allow users immediate access to the CPU, response time may be impacted. This would probably be perceived as erratic response time, as described above, but extreme cases could produce consistently slow response time. Raising the minimum MPL constraint to a large value - that is, close to the maximum number of ready users - will improve the response time.

3. The dispatching priority scheme may be a problem.

   If short TSO transactions have a dispatching priority lower than other work, response time may be affected. Ensure that at least long TSO transactions are placed at a lower priority than short TSO transactions. Refer to the discussion under Problem 1. If there is work that must have a higher dispatching priority than short TSO commands, consider using time slicing to lessen the impact of that work on the TSO response time.

4. Long-running TSO commands may be interfering with short TSO response.

   Long TSO commands (especially batch-like work) can be detrimental to short TSO response time. Performance group periods should be used to alter domains, objectives, or dispatching priorities as a command ages. The duration values of each period should be examined to ensure that lengthy commands are not monopolizing the TSO domain.

5. Paging overhead may be impacting TSO response.

   If RMF paging activity reports indicate a high pageable system area paging rate compared to the address space paging rate, TSO response may be delayed due to time spent waiting for pages. Frequently referenced modules have a tendency to remain in storage, but moderately referenced routines may be continually paged in and out. Examine the PLPA and consider "fixing" these moderately referenced routines. Also, reevaluation of IEAPAKxx can help reduce the in-storage size and number of page-ins for the PLPA. If TSO response is being delayed by page faults in the private area, and swap data sets are defined for the system, consider using storage isolation to guarantee a swap-in working set size that is a multiple of the swap set size.

Problem 4: IMS Response Time or Transaction Rate Is Too Good.

Detection: The IMS statistical program, or any other method decided upon when specifying installation requirements, will indicate whether this is a problem.

Possible Causes:

1. Dispatching priority is too high.

   Whether IMS is swappable or non-swappable, if its dispatching priority is placed in the APG range and controlled by SRM, other work may be time sliced above the IMS priority, or IMS may be time sliced with a base priority below other work, causing IMS to slow down. If time slicing is being used, change the pattern to decrease the time where IMS has preferred dispatching priority.

2. Storage access is unrestrained.

   If the favorable response time for IMS is a result of IMS maintaining a large working set, the installation might limit the real storage accessibility for IMS by using storage isolation control.

Problem 5: Interactive Response Time Is Not Good Enough.

Detection: The method decided upon when specifying installation requirements indicates if this is a problem.

Possible Causes:

1. Dispatching priority is not high enough.

   - Recheck the dispatching priority specifications in the IPS to see what has a higher priority. If a group is being time sliced above this interactive system, then check to see if the pattern or TUNIT value is keeping that group time sliced too long.
   - If IMS is being time sliced, change the pattern to increase the percentage of time that IMS receives its preferred dispatching priority, or decrease the TUNIT value if response time is erratic.

2. Excessive page-in rate.

   If the interactive application had to wait an excessive number of times for pages to be brought into real storage, use the storage isolation controls to limit the number of page-ins required.

Problem 6: Batch Turnaround Time Is Not Good Enough.

Detection: SMF type 5 and 30 records provide elapsed time information for all jobs. RMF provides average elapsed times. The installation specified the criteria for elapsed times when establishing fixed requirements.

Possible Causes:

1. Job classing may be ineffective.

   If tape merges, file sorts, or other long jobs are run in a job class with jobs that have fast turnaround requirements, they could monopolize the initiators while short jobs wait on the queue for selection. SMF type 5 records provide sufficient information to determine whether jobs are indeed waiting a long time to be initiated. Ensure that jobs are placed in the proper initiator class and that an adequate number of initiators are started to select these classes.

2. TSO or IMS may be using more of the system than planned.

   If this is true, batch work will have less resources available. Either reinvestigate fixed requirements, or refer to appropriate problems listed in this section. If resources are available and the batch domain is consistently at its maximum MPL, raise the maximum.

3. Swapping may be slowing down throughput.

   The initial ISV for the batch performance group should be large enough to allow the majority of short jobs to complete without a swap. This is an unlikely cause of this problem since swapping is a control that executes in a range of seconds, while throughput is measured in minutes or hours. However, if an excess of initiators are started, there may be inordinate waiting times when swapped out.

4. Jobs that don't have a fast turnaround requirement may be interfering with jobs that do.

   Within the batch domain, use objectives to distinguish between the performance needs of various types of work. The properties of slopes, ISVs, and cut-off points can help ensure that turnaround requirements are met. Use the DOBJ keyword to control the service distribution between several batch domains (production and test for example) so that the turnaround requirements can be met.

Optimizing the Use of Control Mechanisms

Problem 7: Swapping May Be Causing Excessive Overhead.

Detection: The RMF paging activity report lists total swap out counts. These counts will indicate the causes contributing to swapping overhead. Some of these can be attributed to SRM, some to applications. Since swapping is a costly control mechanism, the number of unnecessary swaps should be kept at a minimum.

Possible Causes:

1. "Input terminal wait" swap count is high.

   In a TSO environment, the vast majority of swaps should be of this type. This is not a problem. In a system with available real storage, these should appear as logical rather than physical swaps.

2. "Output terminal wait" swap count is above 0. In a system with available real storage, these should appear as logical rather than physical swaps.

   a. TSO commands or CLISTs may produce output so fast that the TIOC does not have sufficient buffers available or sufficient buffers may not have been allocated initially. This causes the address space to be automatically swapped until the output buffers have been emptied. SRM treats this condition as the end of a transaction. If the excess swaps are counted here, the problem may be eliminated by increasing the OWAITHI value in the IKJPRMxx parmlib member for TSO/TCAM or increasing the HIBFREXT value in the TSOKEY00 parmlib member for TSO/VTAM.

   b. Another application-induced practice that may cause an output wait swap is the issuance of a TPUT SVC with the 'HOLD' keyword.

3. "Long wait" swap count is high.

   Applications that expect to wait for long periods of time (WAIT macro with the LONG option, STIMER for greater than 0.5 seconds) will be swapped out. Investigate the possibility of rewriting such applications. An ENQ issued for a resource which is held by a swapped out user will also cause a long wait.

4. "Detected wait" swap count is large.

   These swap outs occur when an address space in storage appears to be idle for a fixed period of time as, for instance, when an initiator finishes with a job and then finds no other available jobs on the queue. If this count is larger than the number of swaps that can be attributed to idle initiators, investigate which jobs are waiting and correct the responsible program(s). In addition, check to see if there is a significant delay in operator response to mounts or WTORs.

5. "Unilateral" swap count is high.

   In a batch environment, this count may equal the number of ended batch jobs. If the count is higher than this, the extra swaps may have been caused by the staging of work through several domains with improper minimum MPL values.

   For example, if medium TSO transactions are switched from a domain with a high target MPL into a domain with minMPL = maxMPL = 1, only one medium transaction will be allowed in storage at a time. The others will be swapped out because the target MPL is exceeded, that is, unilateral swap outs will occur.

   The RMF workload activity report indicates, by performance group period, the ratio of swaps to ended transactions. This can assist in finding the period causing extra swapping.

   Note, however, that the technique of delay for TSO as described under Problem 1 will not include unilateral swaps because the delay is incurred in the first period, before the TSO address space is swapped into storage.

6. "Exchange on recommendation value" swap count is high.

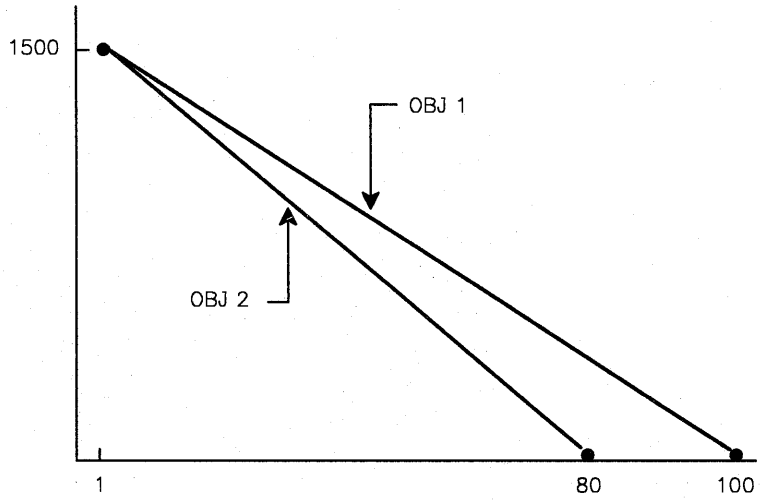
   These control swaps are dictated by the IPS and may be excessive for several reasons.

   a. An ISV value is too small.

      Remember that the ISV is the primary determinant of swapping frequency. It should be large enough to accommodate transactions with such severe response time requirements that swapping is an inappropriate control. Also remember that, to avoid an exchange swap in a period, the ISV specified for that period should be the sum of the durations for that period plus the durations of the preceding periods. RMF workload reports that summarize by domain and period will help locate the area causing the extra swapping.

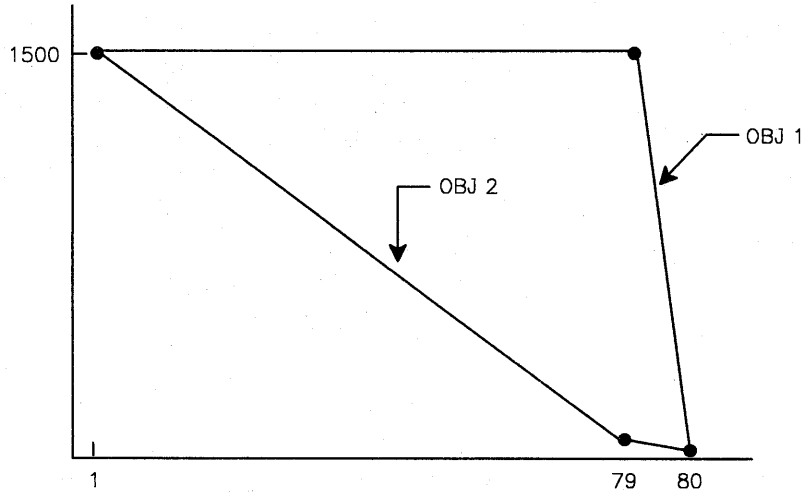
b. Different cut-off workload levels exist in a domain.

The use of different cut-off workload levels is a valid workload management technique. It provides a simple priority scheme within a domain. However, address spaces associated with the objective having the higher cut-off point may be constantly preempting address spaces on other objectives, even in their ISV. If this preemption is frequent, there may be a problem in domain definition. For example:



In a batch domain, a "hot" job on objective 1 will preempt normal jobs on objective 2, and cause an exchange swap. If an installation has many hot jobs, however, they may be candidates for a domain of their own to prevent interference with normal batch work.

Another solution would be to use the same cut-off workload levels, but utilize objective slopes to indicate job priority. Now the hot job on objective 1 will not preempt normal batch until the latter leaves its ISV, after which the hot job will be swapped in and execute to completion.



c. "ENQ Exchange" swap out count is not close to zero. High contention for serial resources is causing extra swapping.

   Even if this count is low, there may still be an enqueue bottleneck if the "unilateral swap count" in a batch environment is much higher than the number of ended batch transactions. This is possible because SRM will exceed the target MPL, if necessary, to ensure that a resource holder is swapped in for its ERV time. Investigate enqueue contention in the system.

Problem 8: Paging Overhead Is High.

Detection: The RMF paging activity report shows the total system non-swap page-in and page-out values. If the sum of these values indicates that more than 5% of CPU time is used for paging, the installation is probably incurring excessive overhead. Five percent is a rate of approximately 150 pages per second for a 3081 Model D, and approximately 210 pages per second for a 3081 Model K. (For other processor model approximations, see the PAGERT1 parameter of parmlib member IEAOPTxx in Section 5.)

In addition to this report, RMF traces the highest system UIC and the count of available frames. A UIC value that is consistently 0 could indicate an over commitment of storage since this means that no pages are allowed to go unreferenced for 1 second. This condition defeats SRM's storage management algorithm, that is, no distinction between the age of frames is possible.

Possible Causes:

1. Over commitment of real storage, probably due to large minimum MPL values.

   SRM will adjust each domain's target MPL based on system contention indicators and the constraint values. If the paging rate indicates the need to decrease the total MPL, but all target MPLs are currently at the minimum MPL values, SRM cannot respond. The installation should either increase the amount of real storage, or decrease the minimum MPL value for at least one domain.

2. Examine the use of the link pack area (LPA). Investigate the feasibility of packing and repackaging to obtain more efficient use of storage.

3. Analyze SMF type 4/5 records. Determine whether certain programs are consistently experiencing or causing high paging/swapping rates.

Problem 9: Resources are Unused.

Detection: RMF reports indicate system resource usage. If these show a high CPU wait time, little paging, or low channel activity, the system may not be fully utilized.

Possible Causes:

1. The maximum MPL for domains is too low.

   If all domains have a target MPL equal to the maximum value, SRM is unable to increase system utilization by raising the target MPL of a domain. Therefore, resources will be under utilized. Increasing the maximum MPL of at least one domain that consistently has ready work waiting should alleviate this.

2. Some resource is a bottleneck.

   If RMF indicates several under utilized resources, but shows one to be consistently over utilized, increasing the MPL of a domain will probably not help. Instead, the service definition coefficients may be altered by raising the coefficient of the over utilized resource. This will have the effect of increasing the service rate for users of that resource, thus producing a lower workload level and causing earlier swap outs.

   Note: Changing the service definition coefficients might require changing other service dependent parameters in the IPS (ISV, DUR, and OBJ). Also, RMF and SMF transaction and service statistics will differ from previous reports. If the over utilized resource can be associated with a specific performance group or period, the load balancing functions can be used to allow swap recommendations for the period on performance group to be modified to eliminate the bottleneck. The RTB keyword in the IPS specifies the load balancing options. In conjunction with these changes, the resource load balancing coefficient of the over utilized resource may need to be raised to give more weight to the load balancing swap recommendations.

3. Not enough initiators have been started.

Section 4: Installation Management Controls

This section contains information about JCL statements and operator commands for SRM-related installation management functions.

The PERFORM Parameter

The PERFORM parameter specifies the control performance group that is assigned to a job, job step, started task, or TSO session only when any one of the following is true:

- There is no IEAICSxx in effect and, for TSO users, the user attribute data set (UADS) permits the PGN to be assigned.
- The installation control specification omits the applicable subsystem (STC, TSO, or job entry subsystem) and, for TSO users, the user attribute data set (UADS) permits the PGN to be assigned.
- The installation control specification includes the subsystem but allows optional performance groups. In this case, the PERFORM value is used if it matches one of the optional performance groups. Again, for TSO, the UADS must permit the PGN, or TSO Extensions must be installed.

The value specified in the PERFORM parameter must be between 1 and 999 and the specified PGN must be defined in the IPS. If an invalid performance group is specified, a warning message indicates that the performance group is not valid and that the system has substituted a default. The default for non-TSO jobs is 1; for TSO it is 2.

The PERFORM parameter assigns the control performance group as follows:

- If PERFORM is specified for a procedure, the specified value is effective for the entire procedure. If PERFORM procstepname is coded for a procedure, the value is effective only for the procedure step named.

If a program is listed as privileged in the program properties table, the PERFORM value for that job or step is ignored. Because it is privileged, the address space is not swapped out except for a long wait. SRM assigns the privileged job or step to a special performance group (0) and domain (0) to ensure that it is in storage when it is needed.

- If no PERFORM parameter is specified on either the JOB or EXEC statements, a default is assigned. If JES2 is the primary job entry subsystem, the installation can specify the default values in the JES2 initialization parameters. (See the parameters &STC, &TSO, &X in the publication *SPL: JES2 Initialization and Tuning*. Defaults can be assigned for started tasks, time sharing users, or any batch class.)
- If a performance group is not specified via the PERFORM parameter and is not assigned by JES2, the IBM-supplied default will be assigned.

- For TSO users, the PERFORM parameter can also be specified on the LOGON command. (See the publication *TSO Command Language Reference.*) If an IEAICSxx parmlib member is in effect at LOGON in a TSO environment that has installed TSO Extensions, the PERFORM parameter is verified by the parmlib member. If no installation control specification is in effect, or the current IEAICSxx member does not contain a TSO section, the UADS is used to verify, and override if necessary, the PERFORM parameter. If the user requests a performance group not specified in the UADS, the IBM-supplied default of 2 is assigned.

For more information on the PERFORM parameter and performance group assignment, see "Installation Control Specification Concepts" earlier in this publication.

Assigning Dispatching Priorities

The DPRTY parameter, which may be included on the EXEC JCL statement, is used to assign a dispatching priority to a job step. (See the publication *JCL Reference.*) Job steps that do not include the DPRTY parameter on their EXEC statements are automatically assigned an APG priority.

If the job step is assigned an APG priority or requests a dispatching priority that falls within the installation defined APG range, the specific priority value is obtained from the IPS specification, if you specify APGRNG=(0-15) in the IPS. If no value is specified in the IPS for the corresponding period, the default is the lowest mean-time-to-wait group.

Operator Commands Related to SRM

The system operator can directly influence SRM's control of specific jobs or groups of jobs by entering commands from the console. The exact format of these commands is defined in the publication *Operations: System Commands.*

The RESET command is used to change the control performance group of an executing job. The new performance group applies to all steps of the job and overrides the performance group assignment in IEAICSxx.

The SETDMN command provides a way for altering the constraint values on a domain's multiprogramming level. The information from this command is valid only for the life of the IPL; it does not change fields in the IPS member.

The SET command with the IPS, ICS, or OPT parameter is used to switch to a different IPS, installation control specification, or OPT after an IPL. SRM will base all control decisions for existing and future jobs on the parameters in the new parmlib member.

The SET IPS and SET ICS functions examine all transactions in the system and change them to the new parameters. If necessary, new transactions are started. See "Installation Control Specification Concepts" and "IPS Concepts" for more information.

To understand whether the system is running effectively with the current IPS, an installation needs dynamic system status information. The DISPLAY command with the DOMAIN keyword provides a snapshot of an individual domain's status.

Section 5: SRM Parameters

This section describes the syntax rules, keywords, and the value ranges and defaults for the parameters specified in the parmlib members IEAIPSxx, IEAICSxx, and IEAOPTxx.

Syntax Rules

Keywords, appearing in upper case letters, must be written exactly as indicated. No imbedded blanks are permitted in the keywords, and the keywords may not span logical records; however, continuation across logical records may occur at any blank or comma, even within operands. Information is written on 80 byte logical records. The data may extend from byte 1 through byte 71; the contents of bytes 72 through 80 will be ignored.

Many of the keywords are used to specify the function's low and high threshold values. For these keywords the syntax is as follows:

keyword=(a,b)

where a is the low threshold value and b is the high threshold value.

Any number of blanks may follow a keyword. All required keywords must be specified in the sequence shown, except where noted, and must be separated by a delimiter. A delimiter, shown in the syntax descriptions as a comma, can be a comma followed by one or more blanks, or it can be any nonzero number of blanks.

Comments are permitted whenever blanks are allowed. The general format of a comment is:

/*character string*/

The slash and asterisk must be immediately adjacent. The character string may contain any characters except the */ combination.

You normally specify SRM keywords from one record to the next by separating the keywords by a comma or blank and specifying the next keyword on the next card image. The following examples illustrates this continuation.

- Card image 1: keyword1=(a,b), keyword2=(c,d)
- Card image 2: keyword3=(x,y,z)
- Card image 3: keyword4=(s,t)

IEAIPSxx Parameters

The IPS contains six categories of information. They should be specified in the order shown.

1. Service definition coefficients:

   [CPU=xx.x] [IOC=xx.x] [MSO=xx.x] [SRB=xx.x]

   Note: These keywords can be specified in any order.

2. General Control Keywords:

   [APGRNG= { x / (x-x) }]

   [PVLDP= { Mx / Fx / Fxy }]

   [TUNIT=xx]

   [TSPTRN=(xx,xx,xx,...)]

   [IOQ= { FIFO / PRTY }]

   [CPGRT=(a,b)]

   [CWSS=(a,b)]

   [IOSRVC= { TIME / COUNT }]

   Note: These keywords, with the exceptions of IOQ, CPGRT, and CWSS, may not be specified if APG is specified in any performance group period definition.

3. Workload levels:

   WKL=(xxx,xxx[,...])

   The maximum number of workload levels that can be specified per IPS is 32.

4. Performance objectives:

   [OBJ=xx,SRV=(xxxx[,...])]

   The maximum number of performance objectives that can be specified per IPS is 64. This set of keywords is coded once for each performance objective.

5. Domains:

   [DMN=xxx [,CNSTR=(xxx,xxx,xxx) / ,CNSTR=(xxx,xxx) [,DOBJ=xx / ,AOBJ=xx / ,FWKL=xxx]]]

   The maximum number of domains that can be specified per IPS is 128. This set of keywords is coded once for each domain.

6. Performance groups:

```
PGN=xxx,([DMN=xxx] [,OBJ=xx] [,RTO=xxx.x],DUR=xxxxxxxx

[ ,UNT= { S } ]   [,ISV=xxxxxx],   [ RTB=x          ]
        { R }                      [ RTB=(x[,...]) ]

[ [ ,APG=xx                                                    ]
  [ ,DP= { Mx } [ ,TSDP= { Mx } [ ,TSGRP=xx ] ] ] [ ,IOP= { Mx } ] ]
         { Fx }          { Fx }                          { Fxy }
         { Fxy }         { Fxy }

[,PPGRT=(a,b)]

[,PPGRTR=(a,b)]

[,PWSS=(a,b)])

[(...)]
```

The keywords within the parentheses represent a performance group period, and are repeated for each period in the performance group. The keywords can appear in any order. The maximum number of periods that can be specified for each performance group is eight.

The PGN keyword must appear once for each performance group. The maximum number of performance groups that can be specified in an IPS is 999.

Parameter Descriptions

Keyword	Meaning	Value Range	Default Value
AOBJ	Specifies a number that associates a domain with a valid performance objective. This performance objective is used to control the average service rate to ready users in this domain. **Example:** OBJ=2,SRV=(100,100) DMN=5,CNSTR=(...,...),AOBJ=2 This indicates that domain 5 should receive 100 service units per second times the number of ready users in the domain. **Restriction:** This keyword is incompatible in the same IPS with the form of CNSTR that specifies a weighting factor.	1-64	None
APG	Specifies the dispatching priority within the APG range that will be assigned to a transaction. The APG is divided into two distinct groups: Mean-time-to-wait 0-6 Fixed 7-15 **Example:** APG=12 The resultant dispatching priority, assuming the default APG range, as described in IEASYSxx, is 7C. **Note:** APG can be used to assign dispatching priority control. Users are encouraged to use the APGRNG and DP keywords rather than APG. **Restriction:** This keyword is not compatible with the following keywords: APGRNG, DP, FRQ, IOP, PVLDP, TSGRP, TSDP, TSPTRN, and TUNIT.	0-15	6
APGRNG	Defines the range of contiguous dispatching priorities that are to be controlled via the IPS. The range is specified as one or more sets of 16 priorities each. The syntax is as follows: APGRNG= { x / [apglo-apghi] } where x apglo, and apghi are integers from 0-15. Apglo and apghi define the low and high dispatching sets in the APG, where apghi > apglo; x defines a single dispatching set. **Example:** APGRNG=(9-11) Defines the APG range to include all job steps whose DPRTY value is between (9,0) and (11,15). **Restriction:** This keyword is incompatible, in the same IPS, with the specification of the APG keyword in a performance group period definition.	0-15	None

Keyword	Meaning	Value Range	Default Value
CNSTR	Specifies the constraints of a domain, that is, the minimum MPL, the maximum MPL, and the contention index control. If the domain definition uses a weighting factor, the syntax is: CNSTR=(a,b,z) where: a = minimum MPL b = maximum MPL z = weighting factor The weighting factor is optional. If the CNSTR keyword is omitted from the definition of a domain, the default values explained under the DMN keyword are used. **Example:** DMN=2, CNSTR=(7,10,15). This indicates for domain number 2, a minimum MPL of 7, a maximum MPL of 10, and a weighting factor of 15. If the domain specification includes a target control keyword and not a weighting factor, the syntax is: CNSTR=(a,b) where a and b are the same as above. **Example:** DMN=5,CNSTR=(2,250),AOBJ=... For domain 5 this indicates a minimum MPL of 2, a maximum MPL of 250 and no weighting factor because the target MPL control is specified by the AOBJ keyword. **Note:** If a domain is specified with neither a weighting factor nor a target control keyword, the following default is used: • If any domain in the IPS uses a target control word, a default of FWKL=1 is assumed. • Otherwise, a weighting factor of 1 is used. **Restriction:** The specification of a weighting factor is incompatible in the same IPS with a target control keyword (AOBJ, DOBJ, or FWKL).	0-999 0-999 1-255	None None (see Note)
CPGRT	Specifies the limits for the common area (common service area and the pageable link pack area) page-in rate. The syntax is as follows: CPGRT=(a,b) where a and b are one to five decimal digits in the range of 1 to 32767 and specify the minimum and maximum acceptable page-ins per elapsed second. The minimum must be less than or equal to the maximum. If the page-in rate is less than the minimum, the target working set size for the common area is decreased. Similarly, if the page-in rate exceeds the maximum, the working set size for the common area is increased. **Example:** CPGRT=(10,30) In this example, if the common area's page-in rate falls below 10 pages per second, the target working set size is decreased thus making additional common area frames eligible to be stolen. If the page-in rate exceeds 30, the target working set size is increased thus making fewer common area frames eligible to be stolen.	a=1-32767 b=1-32767	None None

Keyword	Meaning	Value Range	Default Value
CPU	Indicates the number by which accumulated CPU service units will be multiplied (weighted). **Example:** CPU = 1.0	0.0-99.9	1.0
CWSS	Specifies the range for the target working set size for the common area (common service area, the system queue area, and the pageable link pack area). The syntax is as follows: CWSS = (a,b) where a and b are one to nine decimal digits in the range of 0 to 2147483647 ((2**31)-1) and specify the minimum and maximum target working set size for the common area. The working set is the set of frames allocated to the common area. The minimum must be less than or equal to the maximum. The maximum amount of processor storage is obtained by using an asterisk (*) in place of an explicit value. If you use the asterisk to obtain the storage the storage is (2**31). If the CPGRT keyword is specified, target working set size varies within these limits based on the common area's page-in rate. If CPGRT is not specified, the target is set equal to the minimum specification. **Example 1:** CWSS = (100,200) Assuming that the CPGRT keyword is also specified, the common area's target working set size varies between 100 and 200 frames depending on its page-in rate. **Example 2:** CWSS = (50,100) Assuming that the CPGRT keyword is not specified, the common area's target working set size is fixed at 50 frames. If there is a shortage of real storage, any frames in excess of the specified 100 frames are stolen ahead of frames that are stolen according to the page stealing algorithm. Page stealing does not reduce the working set size below 50.	a = 0-2147483647 b = 0-2147483647	0 the amount of processor storage available
DMN	Specifies a unique number that is used to associate a domain with a transaction. This keyword is used to associate a performance group period with a previously defined domain. If not included in the period definition, the default value is the domain number of the previous period or, if it is the first period, 1. **Example:** PGN = 5,(DMN = 3) In addition, this keyword is used to define a domain, that is to associate the MPL limits and contention index control (specified with the CNSTR keyword) with a unique number. **Example:** DMN = 3,CNSTR = (5,10,1) **Note:** Only domains 0 and 1 are defined by default. Domain 0 is the default domain for privileged jobs (indicated by the privileged bit in the program properties table), and should not be specified in the IPS. It has constraints of (999,999,1) or (999,999),FWKL = 1 if any MPL target control keyword is specified in the IPS. If domain 1 is not specified in the IPS, it will be defined automatically with constraints of (1,999),FWKL = 1 or (1,999,1) if weights are used for other domains in the IPS. Any other domain defined without the CNSTR keyword is given the same defaults.	1-128	(see explanation)

Keyword	Meaning	Value Range	Default Value
DOBJ	Specifies a number that associates a domain with a valid performance objective. This performance objective is used to control the total service rate to a domain. **Example:** DMN=1,CNSTR=(...,...),DOBJ=4 This indicates that domain 1 should receive service at the rate specified in performance objective 4. **Restriction:** This keyword is incompatible in the same IPS with the form of CNSTR that specifies a weighting factor.	1-64	None
DP	Specifies the dispatching control algorithm and dispatching priority for address spaces associated with a performance group period. The syntax is as follows: DP= { Mx / Fx / Fxy } where: M or F defines the dispatching control algorithm, that is, either mean-time-to-wait (M) or fixed (F). x is an integer used to determine the specific APG set within the range specified by the APGRNG parameter (the value of x is added to the value of the lowest set specified by APGRNG). The range for x is: 0 ‰ x ‰ the minimum of (apghi-apglo) or 9 where apglo and apghi are the range values of APGRNG, and 9 is an SRM limit. y is an integer with the range 0-4 that is used to determine the specific fixed priority within the set specified by x, as follows: • the values 0, 1, 2, 3, and 4 of y correspond to the fixed priorities (in hex) B, C, D, E, and F, respectively, within any APG set. • when the Fx syntax is used, it represents a fixed priority (in hex) of A within the APG set specified by x. **Restriction:** This keyword is incompatible in the same IPS with the APG keyword. **Example 1:** DP=F23 This indicates the fixed dispatching control algorithm, and a dispatching priority determined as follows: • Assume APGRNG=(9-11); • Since x=2, the APG set is 9+2=11 or BO-BF (hex); • Since y=3 corresponds to E (hex) within the range of fixed priorities, the actual dispatching priority specified by DP=F23 is BE (hex) or 190 (decimal).	See Explanation	M0

Keyword	Meaning	Value Range	Default Value
	Example 2: DP=F4 This indicates the fixed dispatching control algorithm, and a dispatching priority determined as follows: • Assume APGRNG=(2-9); • Since x=4, the APG set is 2+4=6, or 60-6F (hex); • Since the Fx syntax represents priority 10, or A (hex), within the APG set, the dispatching priority specified by DP=F4 is 6A (hex) or 106 decimal. **Example 3:** DP=M0 This indicates the mean-time-to-wait dispatching control algorithm, and a dispatching priority determined as follows: • Assume APGRNG=7 (note: apghi=apglo); • This means that x must be 0, and therefore the APG set is 7, or 70-7F (hex). • The dispatching priority will be determined by SRM within the mean-time-to-wait range, 70-79 (hex). **Note:** The actual dispatching priority in these examples is not important. It is calculated to explain the syntax descriptions of several IPS keywords. It is only necessary to know the relative hierarchy of dispatching control information.		
DUR	Specifies the length of the performance group period in units indicated by the UNT keyword. DUR should *not* be specified (1) for the last performance period in a performance group, or (2) when there is only one performance period. **Example:** DUR=400 *If UNT=R is specified for a performance group period and a DUR value greater than 1,000,000 is assigned, 1,000,000 is used in place of the assigned value.	0-999999999 or 0-999999K*	None
FWKL	Specifies a workload level that is to be used as a contention index for the domain. **Example:** DMN=3,CNSTR=(...,...),FWKL=100 This indicates that domain 3 will have a contention index of 100. **Restriction:** This keyword is incompatible in the same IPS with the form of CNSTR that specifies a weighting factor.	1-128	1

Keyword	Meaning	Value Range	Default Value
IOC	Indicates the number by which accumulated I/O service units will be multiplied (weighted). **Example:** IOC=1.0	0.00-99.9	1.0
IOP	Specifies the I/O priority of an address space performance group period. The syntax is as follows: IOP= { Mx / Fxy } where the value range for Mx and Fxy is the same as in the DP keyword. (See the DP keyword for more detail). An Mx I/O priority is a fixed priority that is not adjusted within a range as is an Mx dispatching priority. An Fx syntax, specifying the lowest fixed priority, cannot be used with the IOP keyword. **Restrictions:** IOP is meaningful only if IOQ=PRTY is specified in the IPS. The IOP keyword is incompatible in the same IPS with the APG keyword. **Note:** The default value is the address space dispatching priority. For an address space that is time-sliced, the IOP default is the time-slicing dispatching priority (the TSDP value.) If an I/O priority is specified in a mean-time-to-wait range, it is treated as a fixed priority and assigned the lowest value in the specified mean-time-to-wait range. For example, in a mean-time-to-wait range of 70 to 79, the I/O priority is 70. See "I/O Priorities" in the Introduction of SRM Parameter Concepts section for an example.	See explanation	See note
IOQ	Specifies the algorithm to use whenever I/O requests on a device must be queued. **Example:** IOQ=PRTY This indicates that I/O requests are to be queued based on each address space's I/O priority.	FIFO PRTY	FIFO
IOSRVC	Specifies whether I/O service is to be computed using block counts or device connect time. **Example:** IOSRVC=TIME **Note:** If there is a malfunction of the channel measurement facility, device connect time cannot be measured. Instead, block counts are used to compute I/O service.	COUNT TIME	COUNT
ISV	The interval service value specifies the minimum number of service units that a transaction should receive during each interval of real storage occupancy. **Example:** ISV=800	100-999999 or 100-999K	100K
MSO	Indicates the number by which accumulated storage service units will be multiplied (weighted). **Example:** MSO=1.0	0.0-99.9	1.0

Keyword	Meaning	Value Range	Default Value
OBJ	This keyword is used to associate a performance group period with a previously defined performance objective. If the keyword is not included in a period definition, a default value of 1 is assigned. **Example:** PGN=5,(OBJ=3) In addition, this keyword is used to associate the service levels (specified with the SRV keyword) with a unique number in order to define an objective. **Example:** OBJ=3,SRV=(200,100,0) **Note:** Only objective 1 is defined by default. If it is not specified in the IPS, it is defined automatically. The graph of this objective has a very steep slope and a cutoff workload level (that is, where service rate=0) equal to the highest value specified on the WKL parameter.	1-64	(See note)
PGN	Specifies a unique identifier for a performance group definition. This number is used to associate a job, job step, or time-sharing session with the respective performance group. **Example:** PGN=3 **Note:** Each IEAIPSxx member must have performance groups 1 and 2 specified. SRM can assign these default performance groups to address spaces during address space termination. PGN 1 and 2 should be specified in a way so that the work assigned to the PGN is able to execute.	1-999	None
PPGRT	Specifies that execution time is used to calculate the page-in rate for an address space within a performance group period. The syntax is as follows: PPGRT=(a,b) where a and b are one to five decimal digits in the range of 1 to 32767 and specify the minimum and maximum private area page-in rate (non-swap, non- VIO page-ins per accumulated second of execution time). The minimum must be less than or equal to the maximum. If the page-in rate is less than the minimum, the target working set size is decreased thus making additional frames eligible for stealing. Similarly, if the page-in rate exceeds the maximum, the target working set size is increased thus making fewer frames eligible for stealing. **Notes:** 1. The private area page-in rate for a cross memory address space is always based on elapsed time. 2. This keyword is not compatible with the PPGRTR keyword. That is, in a performance group, you can specify either PPGRT or PPGRTR, but not both. **Example:** PPGRT=(5,10) If the address space's private area page-in rate falls below 5 pages per second, the target working set size is lowered. If the page-in rate exceeds 10, the target working set size is raised. The PWSS keyword is used to limit the range of the target working set size.	a=1-32767 b=1-32767	None None

Keyword	Meaning	Value Range	Default Value
PPGRTR	Specifies that residency time is used to calculate the page-in rate for an address space within a performance group period. The syntax is as follows: PPGRTR=(a,b) where **a** and **b** are one to five decimal digits in the range of 1 to 32767 and specify the minimum and maximum private area page-in rate (non-swap, non- VIO page-ins per accumulated second of residency time). The minimum must be less than or equal to the maximum. If the page-in rate is less than the minimum, the target working set size is decreased to make additional frames eligible for stealing. Similarly, if the page-in rate exceeds the maximum, the target working set size is increased to make fewer frames eligible for stealing. **Notes:** 1. The private area page-in rate for a cross memory address space is always based on elapsed time. 2. This keyword is not compatible with the PPGRT keyword. That is, in a performance group, you can specify either PPGRT or PPGRTR, but not both. **Example:** PPGRTR=(5,10) If the address space's private area page-in rate falls below 5 pages per second, the target working set size is lowered. If the page-in rate exceeds 10, the target working set size is raised. The PWSS keyword is used to limit the range of the target working set size.	a=1-32767 b=1-32767	None None
PVLDP	Defines the dispatching algorithm and priority to be assigned to initiators and all other privileged jobs (specified in the program properties table and associated with performance group 0). The syntax is as follows: PVLDP= { Mx - Mean-time-to-wait Fx - Fixed (lowest priority) Fxy - Fixed } For more information, see DP. **Note:** If PVLDP is not coded, initiators and privileged jobs are put in the lowest mean-time-to-wait group. **Example:** PVLDP=F3 **Restrictions:** The priority defined by PVLDP is assigned only to those privileged jobs that do not specify DPRTY or have a DPRTY specified on the //EXEC JCL statement that falls within the APG range. This keyword is incompatible in the same IPS with the APG keyword.	See DP Explanation	M0

Keyword	Meaning	Value Range	Default Value
PWSS	Specifies the range for the target working set size for an address space within a performance group period. The syntax is as follows: PWSS=(a,b) where a and b are one to nine decimal digits in the range of 0 to 2147483647 and specify the minimum and maximum target working set size for an address space. The working set is the set of frames allocated to an address space for its private area and LSQA pages and VIO pages. The minimum must be less than or equal to the maximum. The maximum amount of processor storage is obtained by using an asterisk (*) in place of an explicit value. The target working set size varies within these limits based on the private area page-in rate if the PPGRT or PPGRTR keyword is specified. If you do not specify PPGRT or PPGRTR, the target working set size is equal to the minimum. The minimum value is also used as the minimum swap-in working set size. **Example 1:** PWSS=(10,50) Assuming that PPGRT=(5,10) is specified, the address space's target working set size varies between 10 and 50 frames depending on the private area page-in rate. If the rate falls below 5 frames per second of execution time, the target working set size is lowered. If the page-in rate exceeds 10, the target is raised. **Example 2:** PWSS=(15,30) If there is no PPGRT or PPGRTR specification, the address space's target working set size is fixed at 15. If there is a shortage of real storage, any frames in excess of the specified 30 frames are preferred steal candidates. Page stealing does not reduce the working set size below 15.	a=0-2147483647 b=0-2147483647	0 the amount of processor storage available
RTB	Specifies the response/throughput bias for a particular performance group period. A value of "1" indicates that CPU, I/O, and storage load balancing recommendations are to be added to the workload recommendation during swap analysis. A value of "0" indicates that the CPU, I/O, and stor- age load balancing recommendations are to be ignored. A value of I, C, or S activates the I/O, CPU, or storage load balancing recommendation, respectively. Any combination of the characters I, C, or S is valid, and activates the corresponding load balancing recommendations. **Note:** If more than one value is coded, the list of values must be enclosed in parentheses. **Example:** RTB=(C,I) The CPU and I/O load balancing functions are active for the associated period. **Restriction:** If a list of values is specified for the RTB keyword, no value can be repeated. The values of 0 or 1 must appear alone and cannot be included in the list of values.	0, 1, I, C, or S	0

Keyword	Meaning	Value Range	Default Value
RTO	Specifies the response time (in seconds) for the average first period TSO transaction. **Example:** PGN=2(RTO=2.2) The average TSO transaction that ends during this period receives 2.2 second response time, assuming the system has sufficient capacity. Transactions larger or smaller than the average first period transaction receive a correspondingly longer or shorter response time. **Restrictions:** RTO can be specified only on the first period and is only applicable to TSO transactions. However, the resulting RTO delay increases the response time for all TSO transactions in the performance group, not just those that complete in the first period.	0.0-999.9	0
SRB	Specifies the number by which accumulated SRB service units will be multiplied (weighted). **Example:** SRB=1.0 **Note:** If SRB is not coded, the default is 0.0. The default IPS sets the SRB equal to the CPU coefficient (10.0).	0.0-99.9	(see note)
SRV	Specifies the service rates that define performance objectives. The number of service rates may be equal to or less than the number of workload level values. **Example:** WKL=(10,20,30) OBJ=3,SRV=(100,50,0) **Notes:** 1. If more service rates than workload level numbers are specified, the extra service rates are ignored. 2. An asterisk (*) can appear in place of a numerical SRV value. When it appears between two numerical values (for example SRV=(100,*,0)), it indicates that a linear graph connects the point before the asterisk with the point after the asterisk. Thus, SRV=(100,*,0) is equivalent to SRV=(100,50,0). If no number follows the asterisk, the line joining the previous two values is extended. Thus, SRV=(100,50,*) is equivalent to SRV=(100,50,0). If only one value precedes the asterisk, and no values follow the asterisk, the single value is repeated. Thus, SRV=(100,*) is equivalent to SRV=(100,100,100). If a single value is specified and not followed by an asterisk or another value, the single value is repeated . Thus, SRV=(100) is equivalent to SRV=(100,100,100).	0-999999999	None

Keyword	Meaning	Value Range	Default Value
TSDP	Defines the time slice dispatching control algorithm and dispatching priority for address spaces associated with the performance group period. The syntax is as follows: TSDP= { Mx - Mean-time-to-wait algorithm Fx - Fixed (lowest priority) Fxy - Fixed For more information, see DP. **Example:** TSDP=F6 **Restrictions:** The dispatching priorities that can be specified for DP and TSDP are subject to the following restrictions: • Within the same performance group period, TSDP must specify the same dispatching algorithm (mean-time-to-wait or fixed) as DP and TSDP must be higher than DP. **Example:** DP=F24, TSDP=F64 • It is recommended that for performance concerns the dispatching priority assignments for DP and TSDP be made so that, when a time slice group is raised or lowered, the address spaces within the time slice group maintain the same positions relative to one another. • This keyword is incompatible in the same IPS with the APG keyword in the performance group period definition.	See DP Example	None
TSGRP	Specifies a time slice group number. The specification of TSGRP in the performance group period definition associates a performance group period with a time slice group. The time slice group must appear at least once in the time slice pattern. **Note:** If TSGRP is not specified, the time slice group number is the same as the domain number for the performance group period provided that the domain number is less than 17. If the domain number is greater than 16 you must specify a value for TSGRP. **Example:** TSGRP=2 This indicates that the time slice group number is 2. **Restriction:** This keyword is incompatible in the same IPS with the APG keyword.	1-16	Domain Number (see note)
TSPTRN	Defines the pattern to be followed in distributing time slices to the time slice groups and the percentage of time each group receives preferred dispatching priority. The length of each time slice is 1 SRM time unit. The syntax for TSPTRN is as follows: TSPTRN=(i,j,k,j,...) Where i,j, and k are replaced with time slice group numbers. There may be up to 64 entries in the pattern. The pattern is repeated after the last time slice group specified in the pattern has received its time slice. Each time slice group (TSGRP parameter) in the pattern must be specified in at least one performance group period definition.	1-16	None

Keyword	Meaning	Value Range	Default Value
	An asterisk can replace a time slice group number in TSPTRN to define an interval of time during which no time slice group is given a time slice. **Example:** TSPTRN=(*,1,3,2,3) This indicates that SRM should let one SRM time unit elapse before giving time slice group 1 its time slice. This break occurs each time the time slice pattern is repeated. Also, time slice group 3 receives its high priority 40% of the time (2 of 5 entries in the pattern). **Example:** TSPTRN=(1,2,2,3) This indicates that SRM should distribute time slices in the following order by time slice group number: 1,2,2,3,1,2,2,3,1,2,... **Restrictions:** The following patterns are invalid: TSPTRN=(1) TSPTRN=(1,1,1) (where only one group is specified). This keyword is incompatible in the same IPS with the APG keyword.		
TUNIT	Specifies the number of SRM time units per second. An SRM time unit equals 1 divided by the value specified for TUNIT. The quotient is adjusted by CPU model to allow the IPS to be model independent The duration of one time slice equals one SRM time unit. **Example:** TUNIT=2 This indicates that the number of SRM time units per second is 2. The "SRM Time Unit" thus defined, 0.5 second, is adjusted by CPU mode. **Restriction:** This keyword is incompatible in the same IPS containing the APG keyword.	1-10	1
UNT	Specifies the units for the DUR parameter. "S" indicates service units; "R" indicates real time in seconds. In the following example, the period duration for objective 6 is 4 seconds; for objective 7, 4000 service units. PGN=2,(DMN=1,OBJ=6,DUR=4,UNT=R) (DMN=1,OBJ=7,DUR=4000) (...)	S,R	S
WKL	Specifies a series of positive numbers of increasing value. They provide a reference for defining performance objectives. There can be up to 32 numbers in any workload level specification. The following example illustrates a workload specification and two related performance objectives. WKL=(1,50,99,100) OBJ=3,SRV=(400,300,*,0) OBJ=4,SRV=(400,300,0)	1-128	None

IEAICSxx Parameters

The following section describes the format of the installation control specification and its parameters. The installation control specification consists of multiple subsystem sections. Each subsystem, for which the installation control specification is to assign performance groups, has the following format:

```
SUBSYS=name      ,MASK=n    [ ,PGN=n     [ ,OPGN=n          ]   [ ,RPGN=n] ]
                            [ ,RPGN=n    [ ,OPGN=(n[,n]...) ]              ]

[ { TRXNAME  } { =name     } { ,PGN=n     [ ,OPGN=n          ]   [ ,RPGN=n] } ]
  { USERID   } { =substr(p)}                                                 
  { TRXCLASS }             { ,RPGN=n    [ ,OPGN=(n[,n]...) ]              }
  { ACCTINFO }                                                               
```

Parameter Descriptions

Keyword	Meaning	Value Range	Default Value
MASK = *character*	Specifies a character to match any character in a TRXNAME, USERID, TRXCLASS, or ACCTINFO. This makes possible non-contiguous entry name patterns in the ICS. If MASK is specified, it must be the first ICS keyword. Specify only one MASK keyword for an IEAICSxx member. Once the mask is defined, it is defined for the entire ICS. *character* must be an alphanumeric. *character* can NOT be either of the following: left parenthesis (comma ,	1 character	None
OPGN	Specifies one or more optional control performance groups that the user can request by means of the PERFORM parameter. The control performance must also be defined in the IPS. **Note:** If you specify the OPGN keyword, you must also specify the PGN keyword. You might also want to specify the RPGN keyword, although it is not necessary.	1-999	None
PGN	Specifies a control performance group number. If OPGN is not also specified or if the user-specified performance group is not one of the OPGNs, the performance group specified by the PGN keyword is assigned. A control performance group must also be defined in the IPS.	1-999	None

Keyword	Meaning	Value Range	Default Value
RPGN=*n*	Specifies a performance group that is used only for reporting. A report performance group must not be defined in the IPS. *n* is a number 1-9999.	1-9999	None
substr(p)	A 1 to 8 character string and a column position number. A match will be made with any name containing this string starting in the indicated position. The length of the string plus the column position cannot exceed 9. The string cannot contain the following characters: left parenthesis (comment beginning /* comma , **Example:** D09GES1 matches USERID=GES(4) For more information see "Searching Order for Substrings" in "Installation Control Specification Concepts."	1 to 8 character string	None
SUBSYS=*name*	Specifies the name of a subsystem. Following this are the performance group specifications for transactions associated with this subsystem. The system-defined subsystems are TSO and STC. STC includes all work initiated by the START or MOUNT command. *name*is a 1 to 4 character string; the first character must be alphabetic or national; the remaining characters alphabetic, national, or numeric. The SUBSYS keyword delimits each subsystem section. All TRXNAME, USERID, and TRXCLASS specifications belong to the subsystem which is defined by the preceding SUBSYS specification.	1-4 character string	None
TRXCLASS=*name*	Specifies a class name within a subsystem. Transactions belonging to this class are assigned to the specified performance group or groups. *name* may be 1 to 8 characters	1-8 characters	None
TRXNAME=*name*	Specifies a transaction name within a subsystem Transactions having this name are assigned the specified performance group or groups. *name* may be 1 to 8 characters	1-8 characters	None
USERID=*name*	Specifies a user within a subsystem. Transactions for this user id are assigned the specified performance group or groups. *name* may be 1 to 8 characters	1 to 8 characters	None
ACCTINFO=*string*	Specifies a string of contiguous characters in the accounting information within a subsystem. *string* may be 1 to 8 characters. Transactions using this accounting information are assigned specified performance group(s). Only the account number on the JCL job card is used. By using masking and substringing and ACCTINFO, you can identify the jobs to be assigned a particular performance group. For example, MASK=* and ACCTINFO=D58***(8) tells the system that starting with the 8th position in the accounting fields on the JCL job card, any card with 'D58' followed by 3 characters is a match. Commas and blanks are also counted. Therefore, the following accounting information would be a match. //JOBNAME JOB 'P4512C,D58DAVID,B9212T15,S=U'	1-8 characters	None

IEAOPTxx Parameters

The OPT parameters allow the installation to change many internal SRM constants.

The OPT contains the following categories of information:

1. Resource load balancing coefficients:

   [CPU=x.x]
   [,IOC=x.x]
   [,MSO=x.x]

2. Tape device selection options:

   [,SELTAPE=option]

3. Special options:

   [,CNTCLIST=option]
   [,CNTNSW=option]
   [,DVIO=option]

4. Adjusting constants options:

   - Enqueue residence constants

     [,ERV=xxxxxx]

   - SRM invocation interval constant

     [,RMPTTOM=xxxxxx]

   - MPL adjustment constants

     [,RCCASMT=(xxxxx,yyyyy)]
     [,RCCCPUT=(xxx.x,yyy.y)]
     [,RCCPTRT=(xxxxx,yyyyy)]
     [,RCCUICT=(xxxxx,yyyyy)]
     [,RCCCPUP=(xxx.x,yyy.y)]
     [,RCCMSPT=(xxxxx,yyyyy)]
     [,RCCPDLT=(xxxxx,yyyyy)]
     [,RCCFXTT=(xxxx,yyyy)]
     [,RCCFXET=(xxxx,yyyy)]
     [,PAGERT1=(xxxxx,yyyyy)]
     [,PAGERT2=(xxxxx,yyyyy)]

   - Logical swapping options

     [,LSCTAFQ=(xxxxx,yyyyy)]
     [,LSCTMTE=(xxxxxxx,yyyyyyy)]
     [,LSCTUCT=(xxxxx,yyyyy)]
     [,LSCTFTT=(xxx,yyy)]
     [,LSCTFET=(xxx,yyy)]

- CPU management constants

  [,CCCUTT=(xxx.x,yyy.y)]
  [,CCCSIGUR=xxxxx]

- I/O load balancing constants

  [,ICCLPB(DPSDASD)=(xxx,yyy)]
  [,ICCLPB(NDPSDASD)=(xxx,yyy)]
  [,ICCLPB(TAPE)=(xxx,yyy)]
  [,ICCSIGUP=xxx]

- Storage load balancing constants

  [,MCCSBFCF=xxxxx]
  [,MCCSBSIG=xxxxx]
  [,MCCSBAT=(xxxxx,yyyyy)]
  [,MCCSBST=(xxxxx,yyyyy)]
  [,MCCSBSGP=xxx]
  [,MCCSBINP=xxx]
  [,MCCSBDEP=xxx]
  [,MCCSBFTH=xxx]

- Pageable storage shortage constants

  [,MCCFXEPR=xxx]
  [,MCCFXTPR=xxx]

- Extended storage constants

  [,ESCTPOC(n)=xxxxx]
  [,ESCTPOU(n)=xxxxx]
  [,ESCTSTC(n)=xxxxx]
  [,ESCTSTU(n)=xxxxx]
  [,ESCTSWTC(n)=xxxxx]
  [,ESCTSWTU(n)=xxxxx]
  [,ESCTSWWS(n)=xxxxx]
  [,ESCTVIO=xxxxx]
  [,ESCTVF=xxxxx]

- Selective enablement for I/O constants

  [,CPENABLE=xxx,yyy]

Note: The keywords need not be grouped by category and can appear in any order. The system ignores repetition of a keyword. See "Syntax Rules" at the beginning of Section 5 for the general syntax rules. The system also ignores keywords that were used in prior releases and no longer required, such as RFC and RMC.

Parameter Descriptions

Keyword	Meaning	Value Range	Default Value
CCCUTT	Specifies the low (CCCUTLOT) and high (CCCUTHIT) processor utilization threshold values used in CPU load balancing to determine the degree that the processor is over or underutilized. The syntax is: CCCUTT=(a,b)	a=0.0-128.0 b=0.0-128.0 percent	85 100
CCCSIGUR	Specifies the minimum mean-time-to-wait threshold value for heavy CPU users. Users exceeding this threshold may be considered for CPU load balancing. This constant is also used to determine the range of mean-time-to-wait values which are assigned to each of the ten mean-time-to-wait dispatching priorities. The specified real time value is adjusted by relative processor speed to become SRM time in order to insure consistent SRM control across various processors.	0-32767 milliseconds	45
CNTCLIST	Specifies if the individual commands in a TSO CLIST are treated as separate commands for transaction control. To use this function, TSO Extensions must be installed. The syntax is: CNTCLIST=option where option is either YES or NO. CNTCLIST=NO specifies that the CLIST is treated as a single transaction. CNTCLIST=YES specifies that each command is to be treated as an individual transaction. By specifying CNTCLIST=YES, SRM control of a TSO command becomes the same whether the command is executed explicitly or as part of a CLIST. The RTO parameter, however, does not affect commands within a CLIST, even if they are treated as individual transactions.	NO YES	NO
CNTNSW	Specifies if non-swappable address spaces are to be included in the current MPL of their domain. The syntax is: CNTNSW=option where option is either YES or NO. CNTNSW=NO specifies that they are not to be counted. CNTNSW=YES specifies that they are to be counted in the current MPL. When CNTNSW=YES is specified, the minimum MPL for each domain should allow for the permanently non-swappable address spaces in the domain. The system component address spaces, such as PC/AUTH and ALLOCAS, are non-swappable and are usually in the default domain for started tasks; count these address spaces when setting the minimum for the domain.	NO YES	NO
CPENABLE	Specifies the low (ICCTPILO) and high (ICCTPIHI) threshold values for the percentage of I/O interruptions to be processed through the test pending interrupt (TPI) instruction path in IOS. SRM uses these thresholds to control the number of processors enabled for I/O interruptions. The syntax is: CPENABLE=(a,b)	a=0-100 b=0-100 percentage	10 30

Keyword	Meaning	Value Range	Default Value
CPU	Specifies the weighting factor by which the recommendation of the CPU load balancing factor is to be multiplied when an address space is evaluated for swapping. **Example:** CPU = 3.0	0.0-9.9	0.1
DVIO	Specifies whether directed VIO is to be active in the system or not. The syntax is: DVIO = option where option is either YES or NO. DVIO = YES, the default, specifies that directed VIO is to be active in the system; that is, the NONVIO keyword of the IEASYSxx parmlib member is honored. DVIO = NO specifies that directed VIO is not to be active in the system; the NONVIO parameter of the IEASYSxx parmlib member is ignored.	YES NO	YES
ERV	Specifies the number of CPU service units that the address space is allowed to absorb before being considered for a swap out based on a recommendation value analysis. An address space should not be swapped out during this period if one of the follow conditions is met: • The address space is enqueued on a system resource needed by another address space. • An authorized program in the address space obtains control of the resource (even if another address space does not need that resource) as a result of issuing a reserve for a DASD device which is SHARED. **Example:** ERV = 2 **Note:** SRM determines the execution time equivalent to the specified ERV by multiplying the ERV by the model-dependent time needed to accumulate 1 CPU service unit. In the example given, if an address space can consume 1 service unit in 10 milliseconds, it will be allowed to execute for 20 milliseconds, when it is enqueued on a resource requested by other address spaces, before it will be eligible for swap-out.	0-999999	500
ESCTBDS	Specifies the criteria age (in seconds) at which SRM will add the processor storage for that block-addressable address space to the reserve capacity of the system. **Example:** ESCTBDS = 1000 The hiperspace that exceeds this time is added to the processors reserve capacity.	0-32767	900
ESCTPOC(n)	Specifies the criteria age (in seconds) at which a *changed* page that is to be *paged out* will be sent to extended storage. Specify a value for n to indicate the type of user owning the page, where n can be 0 - nonswappable, common, or privileged users, 1 - all others not 0 or 2, or 2 - TSO users. **Example:** ESCTPOC(0) = 100 Changed paged-out pages are sent to extended storage when the migration age (ma) is greater than the criteria age (ca). The formula is: ma ca	0-32767	0 = 100 1 = 100 2 = 100

Keyword	Meaning	Value Range	Default Value
ESCTPOU(n)	Specifies the criteria age (in seconds) at which an *unchanged* page that is to be *paged out* will be sent to extended storage. Specify a value for n to indicate the type of user owning the page, where n can be 0 - nonswappable, common, or privileged users, 1 - all others not 0 or 2, or 2 - TSO users. **Example:** ESCTPOU(1)=100 Unchanged paged-out pages are sent to extended storage when the migration age (ma) is greater than the criteria age (ca). The formula is: ma ca	0-32767	0=100 1=100 2=100
ESCTSTC(n)	Specifies the criteria age (in seconds) at which a *changed* page that is to be *stolen* will be sent to extended storage. Specify a value for n to indicate the type of user owning the page, where n can be 0 - nonswappable, common, or privileged users, 1 - all others not 0 or 2, or 2 - TSO users. **Example:** ESCTSTC(2)=15 Changed stolen pages are sent to extended storage when the migration age (ma) is greater than the criteria age (ca). The formula is: ma ca	0-32767	0=0 1=20 2=15
ESCTSTU(n)	Specifies the criteria age (in seconds) at which an *unchanged* page that is to be *stolen* will be sent to extended storage. Specify a value for n to indicate the type of user owning the page, where n can be 0 - nonswappable, common, or privileged users, 1 - all others not 0 or 2, or 2 - TSO users. **Example:** ESCTSTU(0)=0 Unchanged stolen pages are sent to extended storage when the migration age (ma) is greater than the criteria age (ca). The formula is: ma ca	0-32767	0=0 1=20 2=15
ESCTSWTC(n)	Specifies the criteria age (in seconds) at which a *changed* page that has been trimmed for a *swap out* will be sent to extended storage. Specify a value for n to indicate the type of user owning the page, where n can be 0 - privileged users, 1 - all others not 0 or 2, or 2 - terminal wait swap users. **Example:** ESCTSWTC(1)=100 Changed swap-out pages are sent to extended storage when the sum of the system-high unreferenced interval count (uic) and the migration age (ma) is greater than the criteria age (ca). The formula is: mc + ma ca	0-32767	0=100 1=100 2=60

Keyword	Meaning	Value Range	Default Value
ESCTSWTU(n)	Specifies the criteria age (in seconds) at which an *unchanged* page that has been trimmed for a *swap out* will be sent to extended storage. Specify a value for n to indicate the type of user owning the page, where n can be 0 - privileged users, 1 - all others not 0 or 2, or 2 - terminal wait swap users. **Example:** ESCTSWTU(2)=60 Unchanged swap-out pages are sent to extended storage when the sum of the system-high unreferenced interval count (uic) and the migration age (ma) is greater than the criteria age (ca). The formula is: uic + ma ca	0-32767	0=100 1=100 2=60
ESCTSWWS(n)	Specifies the criteria age (in seconds) at which a *working set page* that is ready for a *swap out* will be sent to extended storage. Specify a value for n to indicate the type of user owning the page, where n can be 0 - privileged users, 1 - all others not 0 or 2, or 2 - terminal wait swap users. **Example:** ESCTSWWS(0)=100 Working set swap-out pages are sent to extended storage when the sum of the system-high unreferenced interval count (uic) and the migration age (ma) is greater than the criteria age (ca). The formula is: uic + ma ca Terminal wait pages are sent to extended storage when the sum of the system-high unreferenced interval count (uic) and the migration age (ma) minus the think time (tt) is greater than the criteria age (ca). The formula is: (uic + ma) - tt ca	0-32767	0=100 1=100 2=50
ESCTVF	Specifies the criteria age (in seconds) at which a *virtual fetch* page will be sent to extended storage. **Example:** ESCTVF=15 Virtual fetch pages are sent to extended storage when the migration age (ma) is greater than the criteria age (ca). The formula is: ma ca	0-32767	15
ESCTVIO	Specifies the criteria age (in seconds) at which a *virtual I/O* page will be sent to extended storage. **Example:** ESCTVIO=900 Virtual I/O pages are sent to extended storage when the migration age (ma) is greater than the criteria age (ca). The formula is: ma > ca	0-32767	900
ICCLPB (TAPE)	Specifies the low (ICCLPBLO(1)) and high (ICCLPBHI(1)) logical path utilization threshold values for paths used by tape devices. The syntax is: ICCLPB(TAPE)=(a,b)	a=0-100 b=0-100 percent	50 80

Keyword	Meaning	Value Range	Default Value
ICCLPB (NDPSDASD)	Specifies the low (ICCLPBLO(2)) and high (ICCLPBHI(2)) logical path utilization threshold values for paths used by DASD that do not have dynamic reconnection capability. The syntax is: ICCLPB(NDPSDASD)=(a,b)	a=0-100 b=0-100 percent	15 30
ICCLPB (DPSDASD)	Specifies the low (ICCLPBLO(3)) and high (ICCLPBHI(3)) logical path utilization threshold values for paths used by DASD that have dynamic reconnection capability. The syntax is: ICCLPB(DPSDASD)=(a,b)	a=0-100 b=0-100 percent	20 35
ICCSIGUP	Specifies the minimum percentage of I/O connect time on a logical path by a single user for that user to be considered a significant user of I/O resources. Users exceeding this percentage may be considered for I/O load balancing.	0-100 percent	5
IOC	Specifies the weighting factor by which the recommendation of the I/O load balancing function is to be multiplied, when an address space is evaluated for swapping. **Example:** IOC=2.0	0.0-9.9	0.1
LSCTAFQ	Specifies the low (LSCTAFQL) and the high (LSCTAFQH) thresholds for the number of frames on the available frame queue. These thresholds cause the system think time threshold vaule (LSCTMTE) to increase (toward the LSCTMTEH value) or decrease (toward the LSCTMTEL value). The system think time helps to determine if an address space that is in a long, detected, or terminal wait, is to be physically or logically swapped. The syntax is: LSCTAFQ=(a,b)	a=0-32767 b=0-32767 frames	0 300
LSCTFET	Specifies the low (LSCTFETL) and the high (LSCTFETH) percentages of storage that is fixed within the first 16 megabytes; these thresholds cause the logical swap system think time to increase or decrease. The system think time determines if an address space in a long, detected, or terminal wait is logically or physically swapped. The syntax is: LSCTFET=(a,b)	a=0-100 b=0-100 percent	76 82 percent
LSCTFTT	Specifies the low (LSCTFTTL) and the high (LSCTFTTH) percentages of online storage that is fixed or allocated for page-in or page-out. These thresholds cause the logical swap system think time to increase or decrease. The system think time determines if an address space in a long, detected, or terminal wait is logically or physically swapped. The syntax is: LSCTFTT=(a,b)	a=0-100 b=0-100 percent	58 66 percent

Keyword	Meaning	Value Range	Default Value
LSCTMTE	Specifies the low (LSCTMTEL) and the high (LSCTMTEH) think time threshold values. A low and high threshold of zero eliminates logical swapping. A nonzero low threshold forces logical swapping for all terminal wait TSO transactions with think times less than the threshold even when storage is overutilized. **Note:** A swap that is targeted for extended storage (ESTOR) based on the ESCTSWWS keyword may be converted to a logical swap if there is sufficient real storage available. Therefore, when the system is using extended storage, a low and high threshold of zero may not eliminate logical swapping. The syntax is: LSCTMTE=(a,b)	a=0-2147483 b=0-2147483 seconds	0 30
LSCTUCT	Specifies the low (LSCTUCTL) and the high (LSCTUCTH) UIC threshold values that cause the system think time to increase or decrease. The system think time helps to determine if an address space in terminal wait is physically or logically swapped. The syntax is: LSCTUCT=(a,b)	a=0-32767 b=0-32767 seconds	20 30
MCCFXEPR	Specifies the percentage of storage that is fixed within the first 16 megabytes. SRM uses this threshold to determine when a shortage of pageable storage exists because there are too many fixed pages. The syntax is: MCCFXEPR=xxx	0-100 percent	92 percent
MCCFXTPR	Specifies the percentage of online storage that is fixed or allocated for page-in or page-out. SRM uses this threshold to determine when a shortage of pageable storage exists because there is too much paging. The syntax is: MCCFXTPR=xxx **Note:** SRM uses the lesser of the values, (MCCFXTPR x amount of online storage) and (MCCFXEPR x amount of storage that is fixed within the first 16 megabytes) to set the threshold frame count so that it can detect a shortage of pageable storage caused by too much page fixing. In this way, SRM can detect a shortage of pageable storage caused by too much page fixing before or at the same time as a shortage caused by too much paging.	0-100 percent	80 percent
MCCSBAT	Specifies the low (MCCSBATL) and the high (MCCSBATH) available frame count threshold values used in storage load balancing to determine the degree that storage is either over or underutilized. The syntax is: MCCSBAT=(a,b)	a=0-32767 b=0-32767 frames	100 200

Keyword	Meaning	Value Range	Default Value
MCCSBDEP	Specifies the percentage of the minimum significant user frame count threshold (MCCSBSIG) that is subtracted from the significant user frame count threshold when too few address spaces are considered significant uses of storage. **Example:** MCCSBDEP=4 **Note:** In order for MCCSBDEP to be effective, the specified value must be a large enough percentage of MCCSBSIG to result in a value greater than one.	0-100 percent	5
MCCSBFCF	Specifies the scaling factor used to calculate the swap recommendation value for storage load balancing. The number of frames in excess of the significant user threshold is divided by this factor. The syntax is as follows: MCCSBFCF=xxxxx	1-32,767	2
MCCSBFTH	Specifies the percentage of storage that is fixed within the first 16 megabytes. The SRM storage load balancer uses this percentage to determine if storage is underutilized. The syntax is: MCCSBFTH=xxx **Note:** The storage load balancing threshold (MCCSBFTH) should be less than or equal to the logical swap high threshold (LSCTFETH). You should not consider storage underutilized if logical swapping is being limited by its fixed storage threshold.	0-100 percent	76 percent
MCCSBINP	Specifies the percentage of the minimum significant user frame count threshold (MCCSBSIG) that is added to the significant user frame count threshold when too many address spaces are considered significant users of storage. **Example:** MCCSBINP=15 **Note:** In order for MCCSBINP to be effective, the specified value must be a large enough percentage of MCCSBSIG to result in a value greater than one. In addition, setting MCCSBINP to zero results in MCCSBSIG being a constant threshold.	0-100 percent	10
MCCSBSGP	Specifies the target percentage of the number address spaces eligible for storage load balancing that should be considered significant users of storage.	0-100 percent	50
MCCSBSIG	Specifies the minimum significant user frame count threshold for storage load balancing. Users having fewer frames than this limit will never be considered for storage load balancing. Note that SRM adjusts the significant user frame count threshold but will not decrease the threshold below this value. The syntax is: MCCSBSIG=x	0-32,767 frames	40
MCCSBST	Specifies the low (MCCSBSTL) and the high (MCCSBSTH) unreferenced interval count (UIC) threshold values used in storage load balancing to determine the degree that storage is either over or underutilized. The syntax is: MCCSBST=(a,b)	a=1-32767 b=1-32767	10 30

Keyword	Meaning	Value Range	Default Value
MSO	Specifies the weighting factor by which the recommendation value of the storage load balancing function is to be multiplied during swapping evaluation. **Example:** MSO=1.0	0.0-9.9	0.0
PAGERT1	Specifies the low (CCCPRLO1) and high (CCCPRHI1) demand page rate threshold values for a single central processor. These thresholds are used in conjunction with the CPU utilization thresholds (RCCCPUP) and the page delay time thresholds (RCCMSPT) when considering an MPL increase or decrease. The default values for the thresholds are set at IPL time depending on the model number of the processor. The values are:	a=0-32767 b=0-32767 page/second	see explanation

Model	Low Threshold	High Threshold
3090 Model 120E	93	117
3090 Model 120S	99	125
3090 Model 150E	120	151
3090 Model 150S	158	199
3090 Model 170S	196	240
3090 Model 180E	208	262
3090 Model 180S	268	338
3090 Model 200E	208	262
3090 Model 200S	268	338
3090 Model 280E	208	262
3090 Model 280S	268	338
3090 Model 300E	208	262
3090 Model 300S	268	338
3090 Model 400E	208	262
3090 Model 400S	268	338
3090 Model 500E	208	262
3090 Model 500S	268	338
3090 Model 600E	208	262
3090 Model 600S	268	338
4381 Model 91E	94	120
4381 Model 92E	94	120

Note: For single central processor operation, either during IPL, at the time of the SET OPT command, or as a result of a CONFIG CPU command, the values of CCCPRLO1 and CCCPRHI1 are copied into RCCDPRTL and RCCDPRTH, respectively.

The syntax is: PAGERT1=(a,b)

Keyword	Meaning	Value Range	Default Value
PAGERT2	Specifies the low (CCCPRLO2) and high (CCCPRHI2) demand page rate threshold values to be used when two processors are online. These thresholds are used in conjunction with the CPU utilization thresholds (RCCCPUP) and the page delay time thresholds (RCCMSPT) when considering an MPL increase or decrease. The default values, which are set at IPL time, depend on the model number of the processor. The value can be found by multiplying the value listed in the PAGERT1 parameter by 1.7. **Notes:** 1. For a two-way multiprocessing operation, either during IPL, at the time of the SET OPT command, or as a result of a CONFIG CPU command, the values of CCCPRLO2 and CCCPRHI2 are copied into RCCDPRTL and RCCDPRTH, respectively. 2. For greater than two-way multiprocessing, the low and high demand page rate thresholds are each calculated based on the PAGERT1 values specified and the number of processors online. If PAGERT1 is not specified, the default values shown in the PAGERT1 table will be used in the calculations. The formula is as follows: N-way threshold = PAGERT1 threshold x N x .85 (where N is the number of processors online) The syntax is: PAGERT2=(a,b)	a=0-32767 b=0-32767 page/second	see explanation
RCCASMT	Specifies the low (RCCASMTL) and high (RCCASMTH) ASM queue length threshold values that cause the MPL to be increased or decreased. The default values are both 1000 and should cause this parameter to have no effect on MPL adjustment. The syntax is: RCCASMT=(a,b)	a=0-32767 b=0-32767 pages	1000 1000
RCCCPUP	Specifies the low (RCCCPUPL) and high (RCCCPUPH) CPU utilization threshold values that cause the MPL to be increased or decreased. These thresholds are used in conjunction with the page delay-time thresholds (RCCMSPT) and the demand paging rate thresholds (PAGERT1 and PAGERT2) when considering an MPL change. The syntax is: RCCCPUP=(a,b)	a=0.0-128.0 b=0.0-128.0	95.0 98.0
RCCCPUT	Specifies the low (RCCCPUTL) and high (RCCCPUTH) CPU utilization threshold values that cause the MPL to be increased or decreased. The syntax is: RCCCPUT=(a,b)	a=0.0-128.0 b=0.0-128.0 percent	98.0 100.9
RCCFXET	Specifies the low (RCCFXETL) and the high (RCCFXETH) percentages of storage that is fixed within the first 16 megabytes. SRM uses these thresholds to determine if the system MPL needs to be increased or decreased. The syntax is: RCCFXET=(a,b)	a=0-100 b=0-100 percent	82 88 percent

Keyword	Meaning	Value Range	Default Value
RCCFXTT	Specifies the low (RCCFXTTL) and the high (RCCFXTTH) percentages of online storage that is fixed or allocated for page-out or page-in. SRM uses these thresholds to determine if the system MPL needs to be increased or decreased. The syntax is: RCCFXTT=(a,b)	a=0-100 b=0-100 percent	66 72 percent
RCCMSPT	Specifies the low (RCCMSPTL) and high (RCCMSPTH) page delay time threshold values that cause the MPL to be increased or decreased. These thresholds are used in conjunction with the CPU-utilization thresholds (RCCCPUP) and the demand paging rate thresholds (PAGERT1 and PAGERT2). The syntax is: RCCMSPT=(a,b)	a=0-32767 b=0-32767 msec	100 130
RCCPDLT	Specifies the low (RCCPDLTL) and high (RCCPDLTH) page delay time threshold values that cause the MPL to be increased or decreased. The default values are both 1000 and should cause this parameter to have no effect on MPL adjustment. The syntax is; RCCPDLT=(a,b)	a=0-32767 b=0-32767 msec	1000 1000
RCCPTRT	Specifies the low (RCCPTRTL) and high (RCCPTRTH) page fault rate threshold values that cause the MPL to be increased or decreased. The default values are both 1000 and should cause this parameter to have no effect on MPL adjustment. The syntax is: RCCPTRT=(a,b)	a=0-32767 b=0-32767	1000 1000
RCCUICT	Specifies the low (RCCUICTL) and high (RCCUICTH) UIC threshold values that cause the MPL to be increased or decreased. The syntax is: RCCUICT=(a,b)	a=0-32767 b=0-32767 seconds	2 4
RMPTTOM	Specifies the SRM invocation interval. The specified real-time interval is adjusted by relative processor speed to become SRM time in order to ensure consistent SRM control across various processors. The relationship of real time to SRM time for each processor is described in "Dispatching Priorities" in the "IPS Concepts" section.	1000-999999 msec	1000

Keyword	Meaning	Value Range	Default Value
SELTAPE	Specifies the method by which tape devices are selected for allocation from a list of eligible devices. The syntax is: SELTAPE=option where option is one of the following: • NEXT specifies that SRM selects tape devices in ascending order by device number starting with the lowest device number. After the highest device number has been selected, the selection process begins again with the lowest device number. • RANDOM specifies that SRM selects a tape device at random from the list. • FIRST specifies that SRM selects the first tape device in the list. • LOWEST specifies that SRM selects the tape device with the lowest device number. See "Device Allocation" for the method of generating the list of eligible devices from a list of candidate devices.	NEXT RANDOM FIRST LOWEST	NEXT

Section 6: SRM Constants

This section lists internal SRM constants that are not listed in the OPT, but that an installation with unique resource management requirements might want to change. The constants are located in the storage management control table (MCT).

Constant	Related Function	Purpose	Value
MCCASMT1	Auxiliary storage shortage detection	Auxiliary storage shortage threshold	70%
MCCASMT2	Auxiliary storage shortage detection	Critical auxiliary storage shortage threshold	85%
MCCAFCLO	Pageable frame stealing	The number of frames on the available frame queue when stealing begins (used to set RCEAFCLO).	20 minimum
MCCAFCOK	Pageable frame stealing	The minimum number of frames on the available frame queue when stealing ends used to set RCEAFCOK).	40 minimum

Note: The SRM manages the threshold values for MCCAFCLO and MCCAFCOK dynamically.

Index

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

Numerics

Special Characters

MVS/ESA
System Programming Library:
Initialization and Tuning

READER'S
COMMENT
FORM

GC28-1828-1

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Note: ***Copies of IBM publications are not stocked at the location to which this form is addressed. 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.***

Possible topics for comment are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

_____

_____

_____

_____

What is your occupation? _____

How do you use this publication? _____

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

MVS/ESA System Programming Library: Initialization and Tuning

GC28-1828-1 S370-34

Reader's Comment Form

Cut or Fold Along Line

Fold and tape Please Do Not Staple Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D58, Building 921-2
PO Box 950
Poughkeepsie, New York 12602

Fold and tape Please Do Not Staple Fold and tape

Printed in U.S.A.

IBM®

Program Number
5685-001
5685-002

File Number
S370-34

GC28-1828-1

Printed in U.S.A.