z/OS

# MVS Initialization and Tuning Guide

SA22-7591-02

z/OS

# MVS Initialization and Tuning Guide

SA22-7591-02

**Note**

Before using this information and the product it supports, be sure to read the general information under “Notices” on page B-1.

**Third Edition, October 2003**

This is a major revision to SA22-7591-01.

This edition applies to Version 1 Release 4 of z/OS (5694-A01), Version 1 Release 4 of z/OS.e (5655-G52), and to all subsequent releases and modifications until otherwise indicated in new editions.

IBM welcomes your comments. A form for readers' comments may be provided at the back of this document, or you may address your comments to the following address:

International Business Machines Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+845+432-9405
FAX (Other Countries):
Your International Access Code +1+845+432-9405

IBMLink™ (United States customers only): IBMUSM10(MHVRCFS)
Internet e-mail: mhvrcfs@us.ibm.com
World Wide Web: http://www.ibm.com/servers/eserver/zseries/zos/webqs.html

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this document
- Page number or topic related to your comment

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

# Contents

# Figures

# Tables

# About This Document

This document is a preliminary tuning guide for the MVS element of OS/390. The document describes how to initialize the system and how to get improved system performance.

For information about how to install the software products that are necessary to run OS/390, see *z/OS and z/OS.e Planning for Installation*.

## Who Should Use This Document

This document is for anyone whose job includes designing and planning to meet installation needs based on system workload, resources, and requirements. For that audience, the document is intended as a guide to what to do to implement installation policies.

The document is also for anyone who tunes the system. This person must be able to determine where the system needs adjustment, to understand the effects of changing the system parameters, and to determine what changes to the system parameters will bring about the desired effect.

## Where to Find More Information

Where necessary, this document references information in other documents, using the shortened version of the document title. For complete titles and order numbers of the documents for all products that are part of OS/390, see *z/OS Information Roadmap*.

## Information updates on the web

For the latest information updates that have been provided in PTF cover letters and Documentation APARs for z/OS™ and z/OS.e, see the online document at:

`http://www.s390.ibm.com:80/bookmgr-cgi/bookmgr.cmd/BOOKS/ZIDOCMST/CCONTENTS`

This document is updated weekly and lists documentation changes before they are incorporated into z/OS publications.

## Accessing z/OS licensed documents on the Internet

z/OS licensed documentation is available on the Internet in PDF format at the IBM Resource Link™ Web site at:

`http://www.ibm.com/servers/resourcelink`

Licensed documents are available only to customers with a z/OS license. Access to these documents requires an IBM Resource Link user ID and password, and a key code. With your z/OS order you received a Memo to Licensees, (GI10-0671), that includes this key code. [1]

To obtain your IBM Resource Link user ID and password, log on to:

`http://www.ibm.com/servers/resourcelink`

To register for access to the z/OS licensed documents:

1. z/OS.e™ customers received a Memo to Licensees, (GI10-0684) that includes this key code.

1. Sign in to Resource Link using your Resource Link user ID and password.
2. Select **User Profiles** located on the left-hand navigation bar.

**Note:** You cannot access the z/OS licensed documents unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

Printed licensed documents are not available from IBM.

To print licensed documents, you can use the PDF format on either **z/OS Licensed Product Library CD-ROM** or IBM Resource Link .

## Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can access LookAt from the Internet at:
http://www.ibm.com/eserver/zseries/zos/bkserv/lookat/ or from anywhere in z/OS or z/OS.e where you can access a TSO/E command line (for example, TSO/E prompt, ISPF, z/OS UNIX System Services running OMVS).

The LookAt Web site also features a mobile edition of LookAt for devices such as Pocket PCs, Palm OS, or Linux-based handhelds. So, if you have a handheld device with wireless access and an Internet browser, you can now access LookAt message information from almost anywhere.

To use LookAt as a TSO/E command, you must have LookAt installed on your host system. You can obtain the LookAt code for TSO/E from a disk on your *z/OS Collection* (SK3T-4269) or from the LookAt Web site's **Download** link.

# Summary of changes

**Summary of changes**
**for SA22-7591-02**
**z/OS Version 1 Release 4**

The document contains information previously presented in *z/OS MVS Initialization and Tuning Guide*, SA22-7591-00, which supports z/OS Version 1 Release 3.

**New information**

- Beginning with z/OS V1R3, WLM compatibility mode is no longer available. For more information, see xiii.

**Changed information**

- "The Virtual Storage Address Space" on page 1-11 contains 64-bit addressing information.
- "Page Data Set Sizes" on page 2-1 and "Page Data Set Protection" on page 2-2 reflect current standards.
- "CPU and SRB Service" on page 3-59 contains updated SRM constants with processor information.

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Starting with z/OS V1R3, you may notice changes in the style and structure of some content in this document—for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our documents.

**Summary of changes**
**for SA22-7591-01**
**z/OS Version 1 Release 3**

The document contains information previously presented in *z/OS MVS Initialization and Tuning Guide*, SA22-7591-00, which supports z/OS Version 1 Release 1.

**New information**

- Beginning with z/OS V1R3, WLM compatibility mode is no longer available. Accordingly, you can no longer use most of the System Resource Manager functions described in Chapter 3. The IEAICSxx member, the IEAIPSxx member, and most options in the IEAOPTxx member (those options that previously worked in compatibility mode only) are no longer valid. The information has been left here for reference purposes and for use on backlevel systems.

An appendix with z/OS product accessibility information has been added.

**Summary of changes**
**for SA22-7591-00**
**z/OS Version 1 Release 1**

The document contains information also presented in *OS/390 MVS Initialization and Tuning Guide*.

# Chapter 1. Storage Management Overview

To tailor the system's storage parameters, you need a general understanding of the system initialization and storage initialization processes.

This chapter contains the following topics:


For information about the storage management subsystem (SMS), see the DFSMS library.

## 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.
- 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. MVS locates all of the usable central storage that is online and available to the system, and creates a virtual environment for the building of various system areas.

IPL includes the following major initialization functions:

- Loads the DAT-off nucleus into central storage.
- Loads the DAT-on nucleus into virtual storage so that it spans above and below 16 megabytes (except 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.
- In ESA/390 mode, 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.

The system continues the initialization process, interpreting and acting on the system parameters that were specified. NIP carries out 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. For more information about quick starts and warm starts, see *z/OS MVS Initialization and Tuning Reference*.
- Loads modules into the fixed link pack area (FLPA) or the extended FLPA. Note that NIP carries out 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 carries out 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 1-1 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, MVS establishes system component address spaces. MVS 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:

***MASTER*** Master address space

**ABARS, ABARxxxx**
1 to 15 DFSMShsm secondary address spaces to perform aggregate backup or aggregate recovery processing.

**ALLOCAS** Allocation services and data areas

**ANTMAIN** Concurrent copy support

**APPC** APPC/MVS component

**ASCH** APPC/MVS scheduling

**CATALOG** Catalog functions. Also known as CAS (catalog address space).

**BPXOINIT** z/OS UNIX System Services

**CONSOLE** Communications task

**DFM** Distributed File Manager/MVS

**DFMCAS** Distributed File Manager/MVS

**DLF** Data lookaside facility

**DUMPSRV** Dumping services

**HSM** DFSMShsm

**FTPSERVE** FTP server(s); can be user-specified names.

| | |
|---|---|
| **GDEDFM** | For each Distributed File Manager/MVS user conversation that is active, an address space named GDEDFM is created. |
| **GRS** | Global resource serialization |
| **IEFSCHAS** | Scheduler address space |
| **IOSAS** | I/O supervisor, ESCON, I/O recovery |
| **IXGLOGR** | System logger |
| **JES2** | JES2 |
| **JES2AUX** | JES2 additional support |
| **JES2MON** | JES2 address space monitor |
| **JES3** | JES3 |
| **JES3AUX** | JES3 additional support |
| **JES3DLOG** | JES3 hardcopy log (DLOG) |
| **JESXCF** | JES common coupling services address space |
| **LLA** | Link list |
| **NFS** | DFSMS/MVS Network File System address space |
| **OAM** | IBM 3494 and IBM 3495 Tape Library Data Servers |
| **OMVS** | z/OS UNIX System Services |
| **PCAUTH** | Cross-memory support |
| **PORTMAP** | Portmapper function |
| **RASP** | Real storage manager (includes advanced address space facilities support) |
| **RMM** | DFSMSrmm |
| **RRS** | Resource recovery services (RRS) |
| **SMF** | System management facilities |
| **SOM** | SOMobjects |
| **SMS** | Storage management subsystem |
| **SMSVSAM** | VSAM record level sharing |
| **SMXC** | PDSE locking support |
| **SYSBMAS** | PDSE buffering support |
| **TCPIP** | TCP/IP for MVS |
| **TRACE** | System trace |
| **VLF** | Virtual lookaside facility |
| **XCFAS** | Cross system coupling facility |
| **VTAM** | VTAM |
| **WLM** | Workload management |

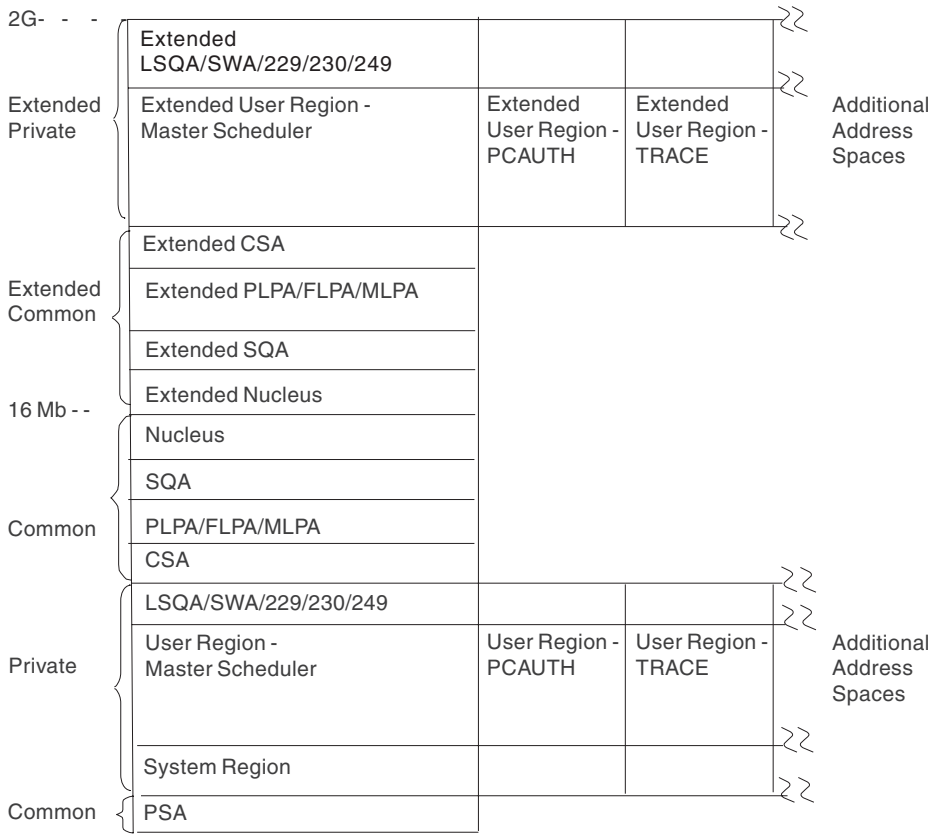



*Figure 1-1. Virtual Storage Layout for Multiple Address Spaces*

## 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. 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. For detailed information about the data contained in IEFSSNxx members for secondary systems, please refer to the installation manual for the specific system.

During system initialization, the defined subsystems are initialized. You should define the primary subsystem (JES) first, because other 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.

**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 *z/OS MVS Initialization and Tuning Reference* for SMS considerations.

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 *z/OS MVS Initialization and Tuning Reference*.

## 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 through START (for batch jobs), LOGON (for time-sharing jobs) or MOUNT, a new address space must be allocated. Note that before LOGON, the operator must have started TCAM or VTAM/TCAS, which have their own address spaces. Figure 1-1 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.

## Processor Storage Overview

Processor storage consists of central storage plus expanded storage. This section provides an overview of central storage. For a discussion of expanded storage, see “Expanded Storage Overview” on page 1-9. Note that expanded storage only applies in ESA/390 mode.

The system uses a portion of both central storage and virtual storage. To determine how much central storage is available to the installation, the system’s fixed storage requirements must be subtracted from the total central storage. The central 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 central storage during initialization and pages in user or system functions for execution. Some RSM functions:

- Allocate central storage to satisfy GETMAIN requests for SQA and LSQA.
- Allocate central storage for page fixing.
- Allocate central storage for an address space that is to be swapped in.
- Allocate and initialize control blocks and queues related to expanded storage.

If there is storage above 16 megabytes, RSM allocates central 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

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

- SQA subpool 226 pages.
- Fixed pages obtained using the RC, RU, VRC, or VRU form of GETMAIN if one of the following is true:
  - LOC=24 is specified.
  - LOC=RES, the default, is either specified or taken, and the program issuing the GETMAIN resides below 16 megabytes, runs in 24-bit mode, and has not requested storage from a subpool supported only above 16 megabytes.
- Fixed pages obtained using the LU, LC, EU, EC, VU, VC, or R form of GETMAIN.
- Storage whose virtual address and real address are the same (V=R pages).

Pages that can reside in central 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 if one of the following is true:
  - LOC=(24,31) is specified.
  - LOC=(RES,31) is specified.
  - LOC=31 is specified.
  - LOC=(31,31) is specified.
  - LOC=RES, the default, is either specified or taken, and the program issuing the GETMAIN resides above 16 megabytes virtual.
  - LOC=RES, the default, is either specified or taken, and the program issuing the GETMAIN resides below 16 megabytes virtual, but runs in 31-bit mode and has requested storage from a subpool supported only above 16 megabytes.
- Any non-fixed page.

**Note:** You may not back the following pages in real storage above 2 gigabytes:

- Nucleus pages.
- SQA pages.
- LSQA pages.

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

The primary requirements/areas composing central 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 called 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 central storage estimates for the page-fixed portions of the installation's programs.

The central storage for the processor, reduced by the global fixed and paged-in virtual storage required to support installation options, identifies the central 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 amount 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, MVS 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 MVS cannot, upon demand, free the frames used for these page types, central 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 central 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 nucleus includes a base and an architectural extension. Specify the correct architectural extension with the ARCHLVL statement in the LOADxx member of SYS1.PARMLIB for your system to run in either ESA/390 mode or z/Architecture mode.

## 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 central 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 any partitioned data set can be included in the FLPA. FLPA is selected through specification of the FIX system parameter in IEASYSxx or from the operator's console at system initialization.

Any module in the FLPA will be treated by the system as though it came from an APF-authorized library. Ensure that you have properly protected any library named in IEAFIXxx to avoid system security and integrity exposures, just as you would protect any APF-authorized library. This area may be used to contain reenterable routines from either APF-authorized or non-APF-authorized libraries that are to be part of the pageable extension to the link pack area during the current IPL.

## 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 central 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 increment and the increment is marked as preferred area storage. An increment is the basic unit of physical storage. For more information about storage increments, see *z/OS MVS Recovery and Reconfiguration Guide*. 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, through the PGFIX macro), 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 central 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 central 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 increment, and the increment 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 Area

This area is used for the execution of job steps specified as fixed because they are assigned to V=R regions in virtual storage (see “Real Regions” on page 1-27). Such jobs run as nonpageable and nonswappable.

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

## Expanded Storage Overview

When running in ESA/390 mode, expanded storage can be thought of as an expansion of central storage. Expanded storage is not used in z/Architecture mode. The purpose of expanded storage is to reduce the paging and swapping of pages between central storage and auxiliary storage, and thus enhance system performance. Because moving a page between central storage and expanded storage is much faster than I/O, use of expanded storage can provide a significant performance advantage.

RSM uses expanded storage as an extension of central storage. When a page is to be removed from central storage, RSM first considers moving it to expanded storage instead of auxiliary storage. When a page that is needed is not in central storage, RSM first checks expanded storage for the page. If the page is in expanded storage, RSM synchronously retrieves the page. If the page is not in expanded 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 expanded storage, see “Expanded storage control” on page 3-49.

When contention for expanded storage increases, the system removes pages from expanded storage to free expanded storage frames. RSM first moves the pages from expanded storage to central 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 the pages are actually sent to auxiliary storage.

## Virtual I/O in Expanded Storage

Expanded storage can be used in ESA/390 mode 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.

**Note:** In ESA/390 mode, expanded storage is used for VIO data set pages, based on demand. Because expanded storage is not used in z/Architecture mode, the VIO data set pages are cached in central storage, based on the availability and demand for central storage.

Expanded 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 expanded storage before all other categories of pages. The criteria age value may be changed by modifying the IEAOPTxx member of SYS1.PARMLIB (for more information, see *z/OS MVS Initialization and Tuning Reference*).

**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 expanded storage is not selective. If VIO is allowed on a system, then any application may choose to use it. If you are using the Storage Management Subsystem (SMS), you can control the use of VIO by creating a VIO storage group. Use the storage group's VIO MAXSIZE attribute to limit the size of the data sets that can use VIO. For more information about using a VIO storage group, see *z/OS DFSMSdfp Storage Administration Reference*.

## Swapping and Migration

RSM is responsible for reclaiming the central storage allocated to an address space when the address space is to be swapped out of central storage. In z/Architecture mode, the swapping to expanded storage and page migration functions are not applicable. RSM is also responsible for building the control structures necessary to efficiently swap the address space back into central storage. When an address space is swapped out of central storage, RSM works with SRM to identify the working set pages that will be swapped back into central storage.

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

If the address space is swapped from central storage to expanded 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 central storage from expanded storage, RSM must return the entire primary working set before the address space can become dispatchable. If the address space is migrated from expanded storage to auxiliary storage, RSM migrates the secondary working set in groups based on the

need to replenish the expanded 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 central 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 central storage in some ratio (for example, 25%). This backup storage contributes significantly to an installation's total central storage requirements.

Virtual storage must also be backed up by expanded storage or auxiliary storage. For information about estimating the amount of storage your installation will need, see the discussion of paging data space in Chapter 2, "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 central storage use and overall system performance. The following overview describes the function of each virtual storage area. For information about identifying problems with virtual storage requests, see "Identifying Problems in Virtual Storage (DIAGxx Parmlib Member)" on page 1-30.

## 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 (SMF), and dumping services
- Each user (batch or TSO/E).

The system uses a portion of each virtual address space.

Each virtual address space consists of:

- The ***common area*** below 16 megabytes
- The ***private area*** below 16 megabytes
- The ***extended common area*** above 16 megabytes
- The ***extended private area*** above 16 megabytes.

Figure 1-2 on page 1-12 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 24-bit, 31-bit, and 64-bit addressing. However, each area and its counterpart above 16 megabytes can be thought of as a single logical area in virtual storage.

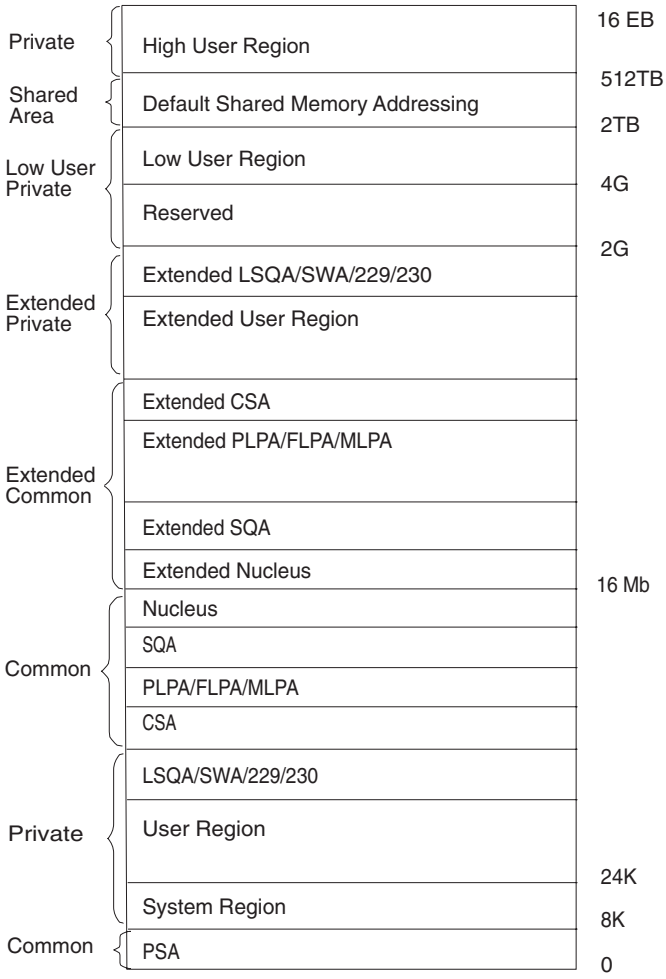



*Figure 1-2. Virtual Storage Layout for a Single Address Space*

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

- Prefixed storage area (PSA).
- 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.

Each storage area in the common area (below 16 megabytes) has a counterpart in the extended common area (above 16 megabytes) with the exception of the PSA.

For more information about using storage above the 2-gigabyte address, see *z/OS MVS Programming: Extended Addressability Guide*.

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, and 249 (the authorized user key area).
- A 16K system region 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 and V=R user regions, 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, the 16K system region area, and subpools 229, 230, and 249) 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 base system control programs require space from each of the basic areas.

Storage for SQA, CSA, LSQA, and SWA is assigned for either the system or a specific user. Generally, space is assigned to 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 and number of private virtual storage address spaces are two 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 through the:
- SQA parameter in the IEASYSxx member of SYS1.PARMLIB
- 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 through the SQA parameter, the system attempts to allocate additional virtual SQA from the CSA area. When certain storage thresholds are reached, as explained below ,the system stops creating new address spaces. When SQA is in use, it is fixed in central storage.

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

### SQA/CSA Shortage Thresholds

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

When the combined total of free SQA + CSA pages falls below the ″high insufficient″ threshold, message IRA100E will be issued. If the number of available SQA and CSA pages falls below the ″low insufficient″ threshold, message IRA101E will issued.

If storage is freed such that the available SQA+CSA amount reaches the ″high sufficient″ threshold, message IRA102I will be issued. The following conditions may be responsible for a shortage of of SQA/CSA:

- There has been storage growth beyond the previous normal range.
- Allocation of SQA and/or CSA is inadequate.
- The current thresholds at which the IRA100E and/or IRA101E messages are issued are too high for your installation.

**Note:** IBM recommends that you do not change the storage thresholds set by the system. Setting the thresholds too low can hamper the ability of the system to recover from storage shortages and may result in unscheduled system outages. Setting the thresholds too high can cause IRA100E, IRA101E, and IRA102I messages to be issued excessively. If you do change the storage thresholds, you should be sure that the threshold is not being reached because of a problem in the system or inadequate allocation of CSA/SQA.

The SQA/CSA threshold levels are contained in the IGVDCLIM CSECT in load module IEAIPL04. The following are the offsets of the threshold values:

*Table 1-1. SQA/CSA Shortage Offset Threshold Values*

| OFFSET | Default Value | System Enforced Minimum | Description | Comments |
|---|---|---|---|---|
| 0000 | x’41000’ | x’9000’ | Sufficient space high | IRA102I message issued at this threshold |
| 0004 | x’21000’ | x’5000’ | Sufficient space low | IRA102I message issued at this threshold |
| 0008 | x’40000’ | x’8000’ | Insufficent space high | IRA100E message issued at this threshold |
| 000C | x’20000’ | x’4000’ | Insufficient space low | IRA101E message issued at this threshold |

If you change the default thresholds, make sure that the new sufficient threshold values are at least x’1000’ larger than the insufficient values. The new values become effective at the next IPL.

## 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. Any module in the

pageable link pack area will be treated by the system as though it came from an APF-authorized library. Ensure that you have properly protected SYS1.LPALIB and any library named in LPALSTxx or on an LPA statement in PROGxx to avoid system security and integrity exposures, just as you would protect any APF-authorized library.

It is desirable to place all frequently used refreshable SYS1.LINKLIB and SYS1.CMDLIB modules in the PLPA because of the following advantages:

- If possible, PLPA is backed by central storage above 16 megabytes; central storage below 16 megabytes is then available for other uses.
- The length of time that a page occupies central storage depends on its frequency of use. If the page is not used over a period of time, the system will reuse (steal) the central storage frame that the page occupies.
- The most frequently used PLPA modules in a time period will tend to remain in central 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 central 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.

See “Placing Modules in the System’s Search Order for Programs” for an alternative suggestion on the placement of some PLPA and SYS1.CMDLIB modules. 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 “The Fixed Link Pack Area (FLPA)” on page 1-7).

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. For more information about IEAPAKxx, see *z/OS MVS Initialization and Tuning Reference* .

## Placing Modules in the System’s Search Order for Programs

Modules (programs), whether stored as load modules or program objects, must be loaded into both virtual storage and central storage before they can be run. When one module calls another module, either directly by asking for it to be run or indirectly by requesting a system service that uses it, it does not begin to run instantly. How long it takes before a requested module begins to run depends on where in its search order the system finds a usable copy and on how long it takes the system to make the copy it finds available.

You should consider these factors when deciding where to place individual modules or libraries containing multiple modules in the system-wide search order for modules:

- The search order the system uses for modules
- How placement affects virtual storage boundaries
- How placement affects system performance

- How placement affects application performance

## The Search Order the System uses for Programs

When a program is requested through a system service (like LINK, LOAD, XCTL, or ATTACH) using default options, the system searches for it in the following sequence:

1. Job pack area (JPA)

   A program in JPA has already been loaded in the requesting address space. If the copy in JPA can be used, it will be used. Otherwise, the system either searches for a new copy or defers the request until the copy in JPA becomes available. (For example, the system defers a request until a previous caller is finished before reusing a serially-reusable module that is already in JPA.)

2. TASKLIB

   A program can allocate one or more data sets to a TASKLIB concatenation. Data sets concatenated to TASKLIB are searched for after JPA but before any specified STEPLIB or JOBLIB. Modules loaded by unauthorized tasks that are found in TASKLIB must be brought into private area virtual storage before they can run. Modules that have previously been loaded in common area virtual storage (LPA modules or those loaded by an authorized program into CSA) must be loaded into common area virtual storage before they can run. For more information about TASKLIB, see *z/OS MVS Programming: Assembler Services Guide*.

3. STEPLIB or JOBLIB

   STEPLIB and JOBLIB are specific DD names that can be used to allocate data sets to be searched ahead of the default system search order for programs. Data sets can be allocated to both the STEPLIB and JOBLIB concatenations in JCL or by a program using dynamic allocation. However, only one or the other will be searched for modules. If both STEPLIB and JOBLIB are allocated for a particular jobstep, the system searches STEPLIB and ignores JOBLIB. Any data sets concatenated to STEPLIB or JOBLIB will be searched after any TASKLIB but before LPA. Modules found in STEPLIB or JOBLIB must be brought into private area virtual storage before they can run. Modules that have previously been loaded in common area virtual storage (LPA modules or those loaded by an authorized program into CSA) must be loaded into common area virtual storage before they can run. For more information about JOBLIB and STEPLIB, see *z/OS MVS JCL Reference*.

4. LPA, which is searched in this order:
   a. Dynamic LPA modules, as specified in PROGxx members
   b. Fixed LPA (FLPA) modules, as specified in IEAFIXxx members
   c. Modified LPA (MLPA) modules, as specified in IEALPAxx members
   d. Pageable LPA (PLPA) modules, loaded from libraries specified in LPALSTxx or PROGxx

   LPA modules are loaded in common storage, shared by all address spaces in the system. Because these modules are reentrant and are not self-modifying, each can be used by any number of tasks in any number of address spaces at the same time. Modules found in LPA do not need to be brought into virtual storage, because they are already in virtual storage.

5. Libraries in the linklist, as specified in PROGxx and LNKLSTxx.

   By default, the linklist begins with SYS1.LINKLIB, SYS1.MIGLIB, and SYS1.CSSLIB. However, you can change this order using SYSLIB in PROGxx and add other libraries to the linklist concatenation. The system must bring modules found in the linklist into private area virtual storage before the programs can run.

**Note:**

1. For more information about which system services load modules, see:
   a. *z/OS MVS Programming: Assembler Services Guide*
   b. *z/OS MVS Programming: Assembler Services Reference ABE-HSP*
   c. *z/OS MVS Programming: Authorized Assembler Services Guide*
   d. *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN*
   e. *z/OS MVS Programming: Authorized Assembler Services Reference ENF-IXG*
   f. *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU*
   g. *z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO*
2. The default search order can be changed by specifying certain options on the macros used to call programs. The parameters that affect the search order the system will use are EP, EPLOC, DE, DCB, and TASKLIB. For more information about these parameters, see the topic about the search for the load module in *z/OS MVS Programming: Assembler Services Guide*.
3. Some IBM subsystems (notably CICS and IMS) and applications (such as ISPF) use the facilities described in the above note to establish other search orders for programs.
4. A copy of a module already loaded in virtual storage might not be accessible to another module that needs it. For example, the copy might reside in another address space, or might have been used or be in use and not be reusable or reentrant. Whenever an accessible copy is not available, any module to be used must be loaded. For more information about the system's search order for programs and when modules are usable or unusable, see the chapter on Program Management in *z/OS MVS Programming: Assembler Services Guide*.

## How Placement Affects Application Performance

Modules begin to run most quickly when all these conditions are true:

- They are already loaded in virtual storage
- The virtual storage they are loaded into is accessible to the programs that call them
- The copy that is loaded is usable
- The virtual storage is backed by central storage (that is, the virtual storage pages containing the programs are not paged out).

Modules that are accessible and usable (and have already been loaded into virtual storage but not backed in central storage) must be returned to central storage from either expanded storage or page data sets on DASD. Modules in the private area and those in LPA (other than in FLPA) can be in virtual storage without being backed by central storage. Because I/O is very slow compared to storage access, these modules will begin to run much faster when they are in central storage or in expanded storage than when they are not. Because data must be copied from expanded storage to central storage, modules can begin to run most quickly when they are in central storage.

Modules placed anywhere in LPA are always in virtual storage, and modules placed in FLPA are also always in central storage. Whether modules in LPA, but outside FLPA, are in central storage depends on how often they are used by all the users of the system, and on how much central storage is available. The more often an

LPA module is used, and the more central storage is available on the system, the more likely it is that the pages containing the copy of the module will be in central storage at any given time.

LPA pages are only stolen, and never paged out, because there are copies of all LPA pages in the LPA page data set. But the results of paging out and page stealing are usually the same; unless stolen pages are reclaimed before being used for something else, they will not be in central storage when the module they contain is called.

LPA modules must be referenced very often to prevent their pages from being stolen. When a page in LPA (other than in FLPA) is not continually referenced by multiple address spaces, it tends to be stolen. One reason these pages might be stolen is that address spaces often get swapped out (without the PLPA pages to which they refer), and a swapped-out address space cannot refer to a page in LPA.

When all the pages containing an LPA module (or its first page) are not in central storage when the module is called, the module will begin to run only after its first page has been brought into central storage.

Modules can also be loaded into CSA by authorized programs. When modules are loaded into CSA and shared by multiple address spaces, the performance considerations are similar to those for modules placed in LPA. (However, unlike LPA pages, CSA pages must be paged out when the system reclaims them.)

When a usable and accessible copy of a module cannot be found in virtual storage, either the request must be deferred or the module must be loaded. When the module must be loaded, it can be loaded from a VLF data space used by LLA, or from load libraries or PDSEs residing on DASD.

Modules not in LPA must always be loaded the first time they are used by an address space. How long this takes depends on:

- Whether the directory for the library in which the module resides is cached
- Whether the module itself is cached in storage
- The response time of the DASD subsystem on which the module resides at the time the I/O loads the module.

The LLA address space caches directory entries for all the modules in the data sets in the linklist concatenation (defined in PROGxx and LNKLSTxx) by default. Because the directory entries are cached, the system does not need to read the data set directory to find out where the module is before fetching it. This reduces I/O significantly. In addition, unless the system defaults are changed, LLA will use VLF to cache small, frequently-used load modules from the linklist. A module cached in VLF by LLA can be copied into its caller's virtual storage much more quickly than the module can be fetched from DASD.

You can control the amount of storage used by VLF by specifying the MAXVIRT parameter in a COFVLFxx member of PARMLIB. You can also define additional libraries to be managed by LLA and VLF. For more information about controlling VLF's use of storage and defining additional libraries, see *z/OS MVS Initialization and Tuning Reference* .

When a module is called, no accessible or usable copy of it exists in central or expanded storage, and it is not cached by LLA, the system must be bring it in from DASD. Unless the directory entry for the module is cached, this involves at least

two sets of I/O operations. The first reads the data set's directory to find out where the module is stored, and the second reads the member of the data set to load the module. The second I/O operation might be followed by additional I/O operations to finish loading the module when the module is large or when the system, channel subsystem, or DASD subsystem is heavily loaded.

How long it takes to complete these I/O operations depends on how busy all of the resources needed to complete them are. These resources include:

- The DASD volume
- The DASD controller
- The DASD control unit
- The channel path
- The channel subsystem
- The CPs enabled for I/O in the processor
- The number of SAPs (CMOS processors only).

In addition, if cached controllers are used, the reference patterns of the data on DASD will determine whether a module being fetched will be in the cache. Reading data from cache is much faster than reading it from the DASD volume itself. If the fetch time for the modules in a data set is important, you should try to place it on a volume, string, control unit, and channel path that are busy a small percentage of the time, and behind a cache controller with a high ratio of cache reads to DASD reads.

Finally, the time it takes to read a module from a load library (not a PDSE) on DASD is minimized when the modules are written to a data set by the binder, linkage editor, or an IEBCOPY COPYMOD operation when the data set has a block size equal to or greater than the size of the largest load module or, if the library contains load modules larger than 32 kilobytes, set to the maximum supported block size of 32760 bytes.

***Access Time for Modules:*** From a performance standpoint, modules not already loaded in an address space will usually be available to a program in the least time when found at the beginning of the list below, and will take more time to be available when found later in the list. Remember that the system stops searching for a module once it has been found in the search order; so, if it is present in more than one place, only the first copy found will be used. The placement of the first copy in the search order will affect how long it takes the system to make the module available. Possible places are:

1. LPA
2. Link list concatenation (all directory entries and some modules cached automatically)
3. TASKLIB/STEPLIB/JOBLIB (with LLA caching of the library)
4. TASKLIB/STEPLIB/JOBLIB (without LLA caching of the library).

For best application performance, you should place as many frequently-used modules as high on this list as you can. However, the following system-wide factors must be considered when you decide how many load modules to place in LPA:

- Performance

  When central storage is not constrained, frequently-used LPA routines almost always reside in central storage, and access to these modules will be very fast.

- Virtual Storage

  How much virtual storage is available for address spaces that use the modules placed in LPA, and how much is available for address spaces that do not use the modules placed in LPA.

## How Placement Affects System Performance

Whether the placement of a module affects system performance depends on how many address spaces use the module and on how often the module is used. Placement of infrequently-used modules that are used by few address spaces have little effect on system-wide performance or on the performance of address spaces that do not use the modules. Placement of frequently-used modules used by a large number of address spaces, particularly those used by a number of address spaces at the same time, can substantially affect system performance.

***Placement of Modules in LPA:*** More central storage can be used when a large number of address spaces each load their own copy of a frequently-used module, because multiple copies are more likely to exist in central storage at any time. One possible consequence of increased central storage use is increased use of expanded storage or increased paging.

When frequently-used modules are placed in LPA, all address spaces share the same copy, and central storage usage tends to be reduced. The probability of reducing central storage usage increases with the number of address spaces using a module placed in LPA, with the number of those address spaces usually swapped in at any given time, and with how often the address spaces reference the module. You should consider placing in LPA modules used very often by a large number of address spaces.

By contrast, if few address spaces load a module, it is less likely that multiple copies of it will exist in central storage at any one time. The same is true if many address spaces load a module, run it once, and then never run it again, as might happen for those used only when initializing a function or an application. This is also true when many address spaces load a module but use it infrequently, even when a large number of these address spaces are often swapped in at one time; for example, some modules are used only when unusual circumstances arise within an application. Modules that fit these descriptions are seldom good candidates for placement in LPA.

You can add modules to LPA in these ways:

- Add the library containing the modules to the LPA list

  Any library containing only reentrant modules can be added to the LPA list, which places all its modules in LPA. Note that modules are added to LPA from the first place in the LPA list concatenation they are found.
- Add the modules to dynamic LPA

  Use this approach instead of placing modules in FLPA or MLPA whenever possible. Searches for modules in dynamic LPA are approximately as fast as those for modules in LPA, and placement of modules in dynamic LPA imposes no overhead on searches for other modules. Note that another LPA module whose address was stored before a module with the same name was loaded into dynamic LPA might continue to be used.
- Add the modules to FLPA

  Modules in the fixed LPA list will be found very quickly, but at the expense of modules that must be found in the LPA directory. It is undesirable to make this list very long because searches for other modules will be prolonged. Note that modules placed in IEAFIXxx that reside in an LPA List data set will be placed in LPA twice, once in PLPA and once in FLPA.
- Add the modules to MLPA

  Like placement in FLPA, placing a large number of modules in MLPA causes searches for all modules to be delayed, and this should be avoided. The delay

will be proportional to the number of modules placed in MLPA, and it can become significant if you place a large number of modules in MLPA.

***Placement of Modules Outside LPA:*** In addition to the effects on central storage mentioned in the previous section, channel subsystem and DASD subsystem load are increased when a module is fetched frequently from DASD. How much it increases depends on how many I/O operations are required to fetch the module and on the size of the module to be fetched.

## How Placement Affects Virtual Storage

When a module is loaded into the private area for an address space, the region available for other things is reduced by the amount of storage used for the module. Modules loaded from anywhere other than LPA (FLPA, MLPA, dynamic LPA, or PLPA) will be loaded into individual address spaces or into CSA.

When a module is added to LPA below 16 megabytes, the size of the explicitly-allocated common area below 16 megabytes will be increased by the amount of storage used for the module. When the explicitly-allocated common area does not end on a segment boundary, IPL processing allocates additional CSA down to the next segment boundary. Therefore, which virtual storage boundaries change when modules are added to LPA depends on whether a segment boundary is crossed or not.

When modules are added to LPA below 16 megabytes, and this does **not** result in the expansion of explicitly-allocated common storage past a segment boundary, less virtual storage will be available for CSA (and SQA overflow) storage. The amounts of CSA and SQA specified in IEASYSxx will still be available, but the system will add less CSA to that specified during IPL.

When the addition of modules to LPA does not result in a reduction in the size of the private area below 16 megabytes, adding load modules to LPA increases the amount of private area available for address spaces that use those load modules. This is because the system uses the copy of the load module in LPA rather than loading copies into each address space using the load module. In this case, there is no change to the private area storage available to address spaces that do not use those load modules.

When modules are added to LPA below 16 megabytes, the growth in LPA can cause the common area below 16 megabytes to cross one or more segment boundaries, which will reduce the available private area below 16 megabytes by a corresponding amount; each time the common area crosses a segment boundary, the private area is reduced by the size of one segment. The segment size in OS/390 is one megabyte.

When the size of the private area is reduced as a result of placing modules in LPA below 16 megabytes, the private area virtual storage available to address spaces that use these modules might or might not be changed. For example, if an address space uses 1.5 megabyte of modules, all of them are placed in LPA below 16 megabytes, and this causes the common area to expand across two segment boundaries, .5 megabytes less private area storage will be available for programs in that address space. But if adding the same 1.5 megabytes of modules causes only one segment boundary to be crossed, .5 megabytes more will be available, and adding exactly 1 megabytes of modules would cause no change in the amount of private area storage available to programs in that address space. (These examples assume that no other changes are made to other common virtual storage area allocations at the same time.)

When the size of the private area is reduced as a result of placing modules in LPA below 16 megabytes, less storage will be available to all address spaces that do **not** use those modules.

A process similar to the one described above is used when ELPA, the other Extended common areas, and the Extended private area are built above 16 megabytes. The only difference is that common storage areas above 16 megabytes are built from 16 megabytes upward, while those below 16 megabytes are built from 16 megabytes downward.

Modules can also be loaded in CSA, and some subsystems (like IMS) make use of this facility to make programs available to multiple address spaces. The virtual storage considerations for these modules are similar to those for LPA.

## Recommendations for Improving System Performance

The following recommendations should improve system performance. They assume that the system's default search order will be used to find modules. You should determine what search order will be used for programs running in each of your applications and modify these recommendations as appropriate when other search orders will be used to find modules.

- Determine how much private area, CSA, and SQA virtual storage are required to run your applications, both above 16 megabytes and below.
- Determine which modules or libraries are important to the applications you care most about. From this list, determine how many are reentrant to see which are able to be placed in LPA. Of the remaining candidates, determine which can be safely placed in LPA, considering security and system integrity.

  **Note:** All modules placed in LPA are assumed to be authorized. IBM publications identify libraries that can be placed in the LPA list safely, and many list modules you should consider placing in LPA to improve the performance of specific subsystems and applications.

  Note that the system will try to load RMODE(ANY) modules above 16 megabytes whenever possible. RMODE(24) modules will always be loaded below 16 megabytes.
- To the extent possible without compromising required virtual storage, security, or integrity, place libraries containing a high percentage of frequently-used reentrant modules (and containing no modules that are not reentrant) in the LPA list. For example, if TSO/E response time is important and virtual storage considerations allow it, add the CMDLIB data set to the LPA list.
- To the extent possible without compromising available virtual storage, place frequently or moderately-used refreshable modules from other libraries (like the linklist concatenation) in LPA using dynamic LPA or MLPA. Make sure you do not inadvertantly duplicate modules, module names, or aliases that already exist in LPA. For example, if TSO/E performance is important, but virtual storage considerations do not allow CMDLIB to be placed in the LPA list, place only the most frequently-used TSO/E modules on your system in dynamic LPA.

  Use dynamic LPA to do this rather than MLPA whenever possible. Modules that might be used by the system before a SET PROG command can be processed cannot be placed solely in dynamic LPA. If these modules are not required in LPA before a SET PROG command can be processed, the library in which they reside can be placed in the linklist so they are available before a SET PROG can be processed, but enjoy the performance advantages of LPA residency later. For example, Language Environment runtime modules required by z/OS UNIX System Services initialization can be made available by placing the SCEERUN

library in the linklist, and performance for applications using Langauge Environment (including z/OS UNIX System Services) can be improved by also placing selected modules from SCEERUN in dynamic LPA.

For more information about dynamic LPA, see the section on PROGxx in *z/OS MVS Initialization and Tuning Reference*. For information about MLPA, see the section on IEALPAxx in the same book.

To load modules in dynamic LPA, list them on an LPA ADD statement in a PROGxx member of PARMLIB. You can add or remove modules from dynamic LPA without an IPL using SET PROG=xx and SETPROG LPA operator commands. For more information, *z/OS MVS Initialization and Tuning Reference* and *z/OS MVS System Commands*.

- By contrast, do not place in LPA infrequently-used modules, those not important to critical applications (such as TSO/E command processors on a system where TSO/E response time is not important), and low-use user programs when this placement would negatively affect critical applications. Virtual storage is a finite resource, and placement of modules in LPA should be prioritized when necessary. Leaving low-use modules from the linklist (such as those in CMDLIB on systems where TSO/E performance is not critical) and low-use application modules outside LPA so they are loaded into user subpools will affect the performance of address spaces that use them and cause them to be swapped in and out with those address spaces. However, this placement usually has little or no effect on other address spaces that do not use these modules.
- Configure as much storage as possible as central storage. Configure only enough expanded storage to support workloads that rely specifically on the availability of ES plus a reasonable additional amount to provide a margin of safety. You can think of the use of ES for paging as if it were very fast I/O. When ES is used for paging, data must be copied from central storage to ES and back again. Before a page in ES can be migrated to auxiliary storage, it must first be copied back into central storage.
- If other measures (like WLM policy changes, managing the content of LPA, and balancing central and expanded storage allocations) fail to control storage saturation, and paging and swapping begin to affect your critical workloads, the most effective way to fix the problem is to add storage to the system. Sometimes, this is as simple as changing the storage allocated to different LPARs on the same processor. You should consider other options only when you cannot add storage to the system.
- If you experience significant PLPA paging, you can use the fixed LPA to reduce page-fault overhead and increase performance at the expense of central storage. You can assure that specific modules are kept in central storage by adding them to the fixed LPA by listing them in IEAFIXxx. This trade-off can be desirable with a system that tends to be CPU-bound, where it can be best to divert some of the central storage from possible use by additional address spaces, and use it for additional LPA modules.

  High-usage PLPA modules probably need not be listed in IEAFIXxx because they tend to be referenced frequently enough to remain in central storage. A large FLPA makes less central storage available for pageable programs. Accordingly, fewer address spaces might be in central storage than would otherwise be the case. No loss in throughput should occur, however, if CPU use remains reasonably high.

  Note that a large FLPA can increase other demand paging and swapping activity, and that this will impede the system's normal self-tuning actions because keeping these modules in storage might prevent other, more frequently-used modules, from being in storage as workloads shift over the course of time. Also, like module packing lists, fixed LPA lists need to be maintained when installing new

releases of software, installing significant amounts of service, or when your workloads change. If you can prevent LPA paging by adding central storage, the system will be simpler to manage.

- When there is significant page-in activity for PLPA pages, and you cannot take other actions to reduce it economically, you can 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 *z/OS MVS Initialization and Tuning Reference*.)

  However, you must maintain module packing lists whenever you install new levels of software or significant service, or when your workloads change. If you can increase the amount of central storage enough to control PLPA paging rather than using a module packing list, the system will be simpler to manage.
- If the first PLPA page data set specified during IPL is large enough, all PLPA pages are written to the same data set. If the first page data set is not large enough to contain all PLPA pages (for example, when allocated as a one-cylinder data set as recommended below), the remaining pages are written to the common page data set (the second one specified during IPL). For best performance, all PLPA pages would be written to a single page data set on a single DASD volume.

  However, a reasonable alternative is to define the PLPA page data set as a single cylinder and define enough storage for the common page data set to contain both the PLPA and common pages. When defined this way, the PLPA and common page data sets should be contiguous, with the small PLPA data set followed immediately by the large common data set on the volume. You should consider allocating these data sets this way unless you experience significant PLPA paging, because it reduces the number of page data sets for which space must be managed and simplifies support.
- If you have significant swapping or paging activity that affects critical applications, and you cannot add storage to manage it, you can tune the paging subsystem.

  When most paging subsystem activity is for swapping, a large number of page data sets can outperform a small number of page data sets, even on high-speed or cached devices. If you have substantial swapping, consider using eight or more page data sets on different low-use volumes on low-use control units and channel paths. However, these should generally be considered stop-gap solutions. If the storage demand continues to grow, tuning the paging subsystem will usually delay the inevitable for only a short time. In the long run, adding central and expanded storage is always a better solution.

  **Note:** Some cached devices also do not support cached paging.

## Modified Link Pack Area (MLPA/Extended MLPA)

This area may be used to contain reenterable routines from either APF-authorized or non-APF-authorized libraries that are to be part of the pageable extension to the link pack area during the current IPL. Any module in the modified link pack area will be treated by the system as though it came from an APF-authorized library. Ensure that you have properly protected any library named in IEALPAxx to avoid system security and integrity exposures, just as you would protect any APF-authorized library.

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.

**Note:** Loading a large number of modules in the MLPA can increase fetch time for modules that are not loaded in the LPA. This could affect system performance.

The MLPA can be used at IPL time 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.
- Including the MLPA system parameter in IEASYSxx or specifying MLPA from the operator's console during system initialization.

## Common Service Area (CSA/Extended CSA)

This area contains pageable and fixed 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 space 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.
- 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 space from extended CSA. When both extended SQA and extended CSA are used up, the system allocates space from SQA and CSA below 16 megabytes. 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.

### SQA/CSA Shortage Thresholds

Ensuring the appropriate size of extended SQA and extended CSA storage is critical to the long-term operation of the system. If the size allocated for extended SQA is too small or is used up very quickly, the system attempts to use extended CSA. When both extended SQA and extended CSA are used up, the system

allocates space from SQA and CSA below 16 megabytes. The allocation of this storage could eventually lead to a system failure.

When the combined total of free SQA + CSA pages falls below the ″high insufficient″ threshold, message IRA100E will be issued. If the number of available SQA and CSA pages falls below the ″low insufficient″ threshold, message IRA101E will issued.

For more information about SQA shortages and the two thresholds, see “SQA/CSA Shortage Thresholds” on page 1-14.

## 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.

LSQA is intermixed with SWA and subpools 229, 230, and 249 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, and 249 downward from 2 gigabytes into the unallocated portion of the extended private area, as needed. (See Figure 1-2 on page 1-12.) 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, and 249.

## Subpools 229, 230, 249 - Extended 229, 230, 249

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 that can be obtained only by authorized programs having appropriate storage protect keys. These control blocks were placed in storage by system components on behalf of the user.

These subpools are intermixed with LSQA and SWA. Subpool 229 is fetch protected; subpools 230 and 249 are not. All three subpools are pageable. Subpools 229 and 230 are freed automatically at task termination; subpool 249 is freed automatically at jobstep 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 called the **user region**.

## Types of User Regions

There are two types of user regions: virtual (or V=V) and real (or V=R). Virtual and real regions are mutually exclusive; private areas can be assigned to V=R or V=V, but not to both. It is the installation's responsibility to determine the type of region to which jobs are assigned. Usually, 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 that require a one-to-one mapping from virtual to central storage, such as program control interruption (PCI) driven channel programs, are candidates for real regions.

Two significant differences between virtual and real regions are:

- How they affect an installation's central storage requirements
- How their virtual storage addresses relate to their central storage addresses.

For virtual regions, which are pageable and swappable, the system allocates only as many central storage frames as are needed to store the paged-in portion of the job (plus its LSQA). The processor translates the virtual addresses of programs running in virtual regions to locate their central storage equivalent.

For real regions, which are nonpageable and nonswappable, the system allocates and fixes as many central storage frames as are needed to contain the entire user region. The virtual addresses for real regions map one-for-one with central storage addresses.

***Virtual Regions:*** Virtual regions begin at the top of the system region (see Figure 1-2) and are allocated upward through the user region to the bottom of the area containing the LSQA, SWA, and the user key area (subpools 229, 230, and 249). Virtual regions are allocated above 16 megabytes also, beginning at the top of the extended CSA, and upward to the bottom of the extended LSQA, SWA, and the user key area (subpools 229, 230, and 249).

As a portion of any V=V job is paged in, 4K multiples (each 4K multiple being one page) of central storage are allocated from the available central storage. Central 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 through JCL (see "Specifying Region Size" on page 1-28).

***Real Regions:*** Real regions begin at the top of the system region (see Figure 1-2) and are allocated upward to the bottom of the area containing LSQA, SWA, and the user key area (subpools 229, 230, and 249). Unlike virtual regions, real regions are only allocated below 16 megabytes.

The system assigns real regions to a virtual space within the private area that maps one-for-one with the real addresses in central storage below 16 megabytes. Central storage for the entire region is allocated and fixed when the region is first created.

***Specifying Region Type:*** A user can assign jobs (or job steps) to virtual or real regions by coding a value of VIRT or REAL on the ADDRSPC parameter on the job's JOB or EXEC statement. For more information on coding the ADDRSPC parameter, see *z/OS MVS JCL Reference*.

The system uses ADDRSPC with the REGION parameter. The relationship between the ADDRSPC and REGION parameter is explained in *z/OS MVS JCL User's Guide*.

## Region Size and Region Limit

***What is Region Size?:*** The region size is the amount of storage in the user region available to the job, started task, or TSO/E user. The system uses region size to determine the amount of storage that can be allocated to a job or step when a request is made using the STORAGE or GETMAIN macros and a variable length is requested. The region size minus the amount of storage currently allocated, determines the maximum amount of storage that can be allocated to a job by any single variable-length GETMAIN request.

***What is Region Limit?:*** The region limit is the maximum total storage that can be allocated to a job by any combination of requests for additional storage using the GETMAIN or STORAGE macros. It is, in effect, a second limit on the size of the user's private area, imposed when the region size is exceeded.

***Specifying Region Size:*** Users can specify a job's region size by coding the REGION parameter on the JOB or EXEC statement. The system rounds all region sizes to a 4K multiple.

The region size value should be less than the region limit value to protect against programs that issue variable length GETMAINs with very large maximums, and then do not immediately free part of that space or free such a small amount that a later GETMAIN (possibly issued by a system service) causes the job to fail.

For V=V jobs, the region size can be as large as the entire private area, minus the size of LSQA/SWA/user key area (subpools 229, 230, and 249) and the system region (see Figure 1-2).

For V=R jobs, the REGION parameter value cannot be greater than the value of the REAL system parameter specified at IPL. If the user does not explicitly specify a V=R job's region size in the job's JCL, the system uses the VRREGN system parameter value in the IEASYS00 member of SYS1.PARMLIB.

For more information on coding the REGION parameter, see *z/OS MVS JCL Reference*.

**Note:** VRREGN should not be confused with the REAL system parameter. REAL specifies the total amount of central 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.

An installation can override the VRREGN default value in IEASYS00 by:

- Using an alternate system parameter list (IEASYSxx) that contains the desired VRREGN parameter value.
- Specifying the desired VRREGN value at the operator's console during system initialization. This value overrides the value for VRREGN that was specified in IEASYS00 or IEASYSxx.

For V=R requests, if contiguous storage of at least the size of the REGION parameter or the system default is not available in virtual or central storage, a request for a V=R region is placed on a wait queue until space becomes available.

Note that V=R regions must be mapped one for one into central 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 central storage area assigned to their region, and to LSQA, SWA, and subpools 229, 230, and 249.

## Limiting User Region Size

***Why Control Region Size or Region Limit?:*** It is the installation's choice whether to control a program's maximum region size or region limit. The region size allowed to users can affect performance of the entire system. When there is no limit on region size and the system uses its default values, users might obtain so much space within a region (by repeated requests for small amounts of storage or a single request for a large amount) that no space would remain in the private area for the system to use. This situation is likely to occur when a program issues a request for storage and specifies a variable length withsuch a large maximum value that most or all of the space remaining in the private area is allocated to the request. If this program actively uses this large amount of space (to write tables, for example), it can affect central storage (also known as real storage) and thus impact performance.

To avoid an unexpected out-of-space condition, the installationshould require the specification of some region size on the REGION parameter of JOB or EXEC JCL statements. Also, the installation can set system-wide defaults for region size through the job entry subsystem (JES) or with MVS installation exit routines. The installation can control user region limits for varying circumstances by selecting values and associating them with job classes or accounting information.

***Using JES to Limit Region Size:*** After interpreting the user-coded JCL, JES will pass to MVS either the value requested by the user on the REGION parameter, or the installation-defined JES defaults. JES determines the REGION value based on various factors, including the source of the job, or the class of the job, or both.

The 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 IBM 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.

For more information on JES defaults, see either *z/OS JES2 Initialization and Tuning Reference* or *z/OS JES3 Initialization and Tuning Reference*, depending on which JES your system is using.

***Using Exit Routines to Limit Region Size:*** Two MVS installation exits are available to override the region values that JES passes to MVS. The exit routines are called IEFUSI and IEALIMIT.

The installation can use either IEFUSI or IEALIMIT to change the job's limit for the region size below 16 megabytes of virtual storage. However, to change the limit for the region size above 16 megabytes, the installation must code the IEFUSI installation exit.

If your installation does not supply an IEFUSI exit routine, the system uses the default values in the IBM-supplied IEALIMIT exit routine to determine region size. To determine region limit, the system adds 64K to the default region size.

For more information on IEFUSI and IEALIMIT, see *z/OS MVS Installation Exits*.

# Identifying Problems in Virtual Storage (DIAGxx Parmlib Member)

This section describes functions you can use to identify problems with virtual storage requests (such as excessive use of common storage, or storage that is not freed at the end of a job or address space). The functions are:

- Common storage tracking
- GETMAIN/FREEMAIN/STORAGE (GFS) trace.

You can control the status of these functions in the DIAGxx parmlib member.

## Using the Common Storage Tracking Function

Common storage tracking collects data about requests to obtain or free storage in CSA, ECSA, SQA, and ESQA. You can use this data to identify jobs or address spaces that use up an excessive amount of common storage or have ended without freeing common storage. If those jobs or address spaces have code to free that storage when they are canceled, you might relieve the shortage and avoid an IPL if you cancel those jobs or address spaces using an operator command.

You can use Resource Measurement Facility (RMF) or a compatible monitor program to display the data that the storage tracking function collects. You can also format storage tracking data in a dump using interactive problem control system (IPCS). For information on how to use IPCS to format common storage tracking data, see the description of the VERBEXIT VSMDATA subcommand in *z/OS MVS IPCS Commands*.

The OWNER parameter on the CPOOL BUILD, GETMAIN, and STORAGE OBTAIN macros assigns ownership of the obtained CSA, ECSA, SQA, or ESQA storage to a particular address space or the system. To get the most value from the common storage tracking function, ensure that authorized programs specify the OWNER parameter on all CPOOL BUILD, GETMAIN, and STORAGE OBTAIN macros that:

- Request storage in CSA, ECSA, SQA, or ESQA, *and*
- Have an owning address space that is *not* the home address space.

IBM recommends that common storage tracking always be activated.

You can turn storage tracking on by activating a DIAGxx parmlib member, either at IPL (specify DIAG=xx as a system parameter) or during normal processing (enter a SET DIAG=xx command). For more information about using SET DIAG=xx, see *z/OS MVS System Commands*.

If you do not specify a DIAGxx parmlib member at IPL, the system processes the default member DIAG00. If DIAG00 does not exist, common storage tracking will not be turned on. Common storage remains active until turned off through a SET DIAG=*xx* command. DIAG00 also turns off the GFS trace function, which is described in “Using GETMAIN/FREEMAIN/STORAGE (GFS) Trace” on page 1-31.

IBM also provides the DIAG01 parmlib member, which turns the common storage tracking function on, and DIAG02, which turns the common storage tracking function off. Your installation must create any additional DIAGxx parmlib members.

### Using GETMAIN/FREEMAIN/STORAGE (GFS) Trace

GFS trace helps you identify problems related to requests to obtain or free storage. GFS trace uses the generalized trace facility (GTF) to collect data about storage requests, and places this data in USRF65 user trace records in the GTF trace. You can use IPCS to format the GTF trace records.

To turn GFS trace on or off, activate a DIAGxx parmlib member, either at IPL (specify DIAG=xx as a system parameter) or during normal processing (enter a SET DIAG=xx operator command). If you do not specify a DIAGxx parmlib member at IPL, the system processes the default member DIAG00, which turns GFS trace off. See *z/OS MVS System Commands* for more information about using SET DIAG=xx.

The DIAGxx parmlib member also allows you to specify:

- **GFS trace filtering data**, which limits GFS trace output according to:
  - Address space identifier (ASID)
  - Storage subpool
  - The length of the storage specified on a request
  - Storage key
- **GFS trace record data**, which determines the data to be placed in each record, according to one or more data items specified on the DATA keyword.

GFS trace degrades performance to a degree that depends on the current workload in the system. Therefore, it is a good idea to activate GFS trace only when you know there is a problem.

For information about how to request a GFS trace, see *z/OS MVS Diagnosis: Tools and Service Aids*.

## 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 1-3):

- System data set storage area.
- Paging data sets for backup of all pageable address spaces.

## 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 installation. 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 *z/OS MVS System Data Set Definition*.

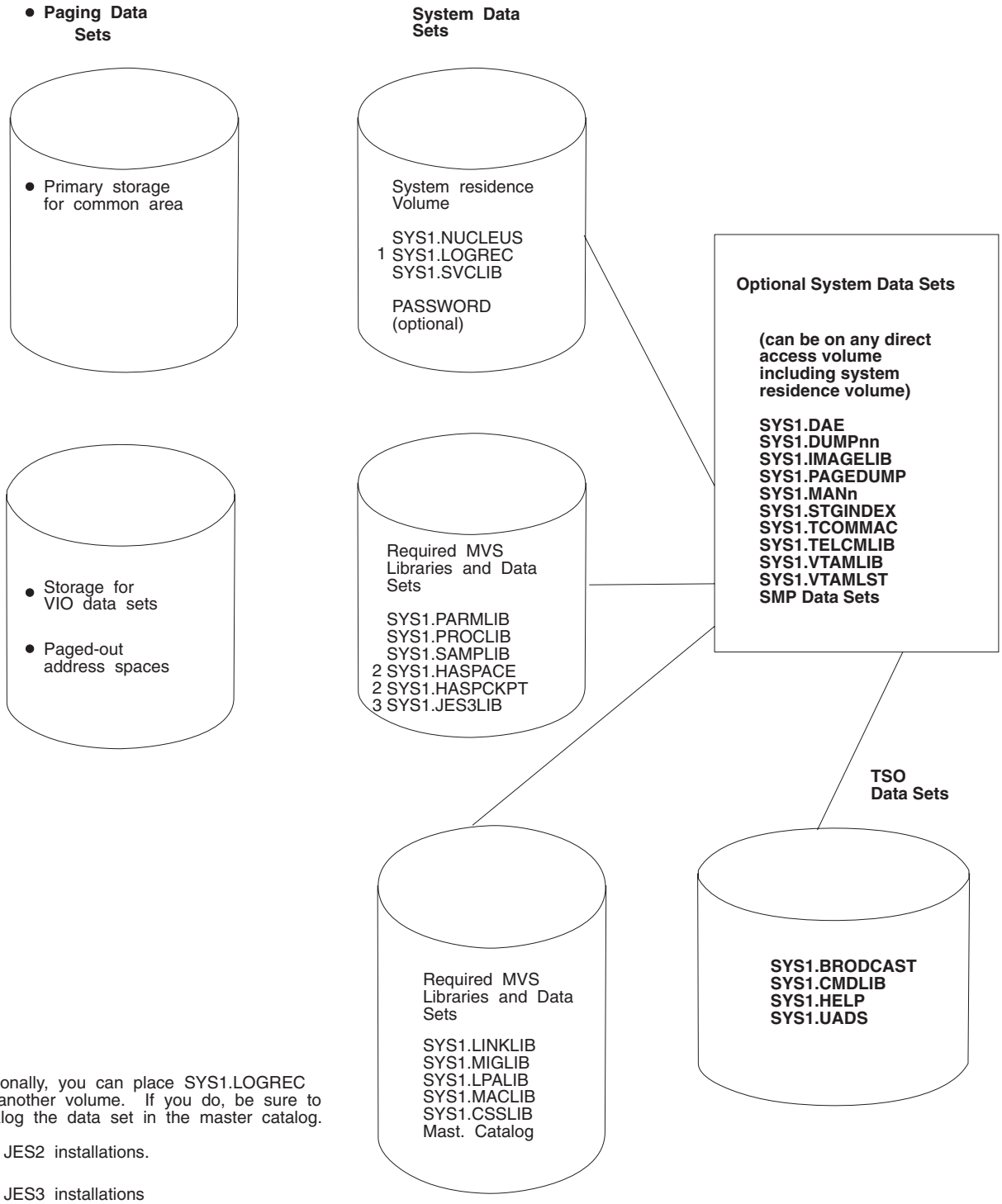



*Figure 1-3. Auxiliary Storage Requirement Overview*

## 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 may be 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.

Paging data sets are 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 installation, 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 reIPL 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.

## 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, with the PAGE parameter in the IEASYSxx parmlib member, allows 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 and the DVIO keyword of IEAOPTxx is explained more fully in *z/OS MVS Initialization and Tuning Reference*.

## Primary and Secondary PLPA

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

On the PLPA page dataset, 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 dataset. CLPA also implies CVIO (see below).

### Virtual I/O Storage Space

Virtual I/O operations may send VIO dataset pages to the local paging dataset space. During a quick start, the installation uses the CVIO parameter to purge VIO dataset pages. During a warm start, the system can recover the VIO dataset 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 dataset pages that are associated with journaled job classes. (The VIO journaling dataset contains entries for the VIO datasets associated with journaled job classes.) If there are no journaled job classes or no VIO journaling dataset, there is no recovery of VIO dataset pages. Instead, all VIO dataset 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 that was defined with having VIO=YES, is not specified, the default size is 10 primary and 50 secondary blocks with an average block length of 1000 bytes.

The cumulative number of page datasets must not exceed 256. This maximum number of 256 page datasets should follow these guidelines:

- There must be exactly one *PLPA* page dataset.
- There must be exactly one *common* page dataset.
- There must be at least one *local* page dataset, but no more than 253.

The actual number of pages required in paging datasets depends on the system load, including the size of the VIO datasets 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.

## Improving Module Fetch Performance With LLA

You can improve the performance of module fetching on your system by allowing library lookaside (LLA) to manage your production load libraries. LLA reduces the amount of I/O needed to locate and fetch modules from DASD storage.

Specifically, LLA improves module fetch performance in the following ways:

- By maintaining (in the LLA address space) copies of the library directories the system uses to locate load modules. The system can quickly search the LLA copy of a directory in virtual storage instead of using costly I/O to search the directories on DASD.
- By placing (or **staging**) copies of selected modules in a virtual lookaside facility (VLF) data space (when you define the LLA class to VLF, and start VLF). The system can quickly fetch modules from virtual storage, rather than using slower I/O to fetch the modules from DASD.

  LLA determines which modules, if staged, would provide the most benefit to module fetch performance. LLA evaluates modules as candidates for staging based on statistics it collects about the members of the libraries it manages (such as module size, frequency of fetches per module (fetch count), and the time required to fetch a particular module).

If necessary, you can directly influence LLA staging decisions through installation exit routines (CSVLLIX1 and CSVLLIX2). For information about coding these exit routines, see *z/OS MVS Installation Exits* .

## LLA and Module Search Order

When a program requests a module, the system searches for the requested module in various system areas and libraries, in the following order:

1. Modules that were loaded under the current task (LLEs)
2. The job pack area (JPA)
3. Tasklib, steplib, joblib, or any libraries that were indicated by a DCB specified as an input parameter to the macro used to request the module (LINK, LINKX, LOAD, ATTACH, ATTACHX, XCTL or XCTLX).
4. Active link pack area (LPA), which contains the FLPA and MLPA
5. Pageable link pack area (PLPA)
6. SYS1.LINKLIB and libraries concatenated to it through the LNKLSTxx member of parmlib. (“Placing Modules in the System’s Search Order for Programs” on page 1-15 explains the performance improvements that can be achieved by moving modules from the LNKLST concatenation to LPA.)

When searching TASKLIBs, STEPLIBs, JOBLIBs, a specified DCB, or the LNKLST concatenation for a module, the system searches each data set directory for the first directory entry that matches the name of the module. The directory is located on DASD with the data set, and is updated whenever the module is changed. The directory entry contains information about the module and where it is located in storage. (For more information, see the “Program Management” topic in the *z/OS MVS Programming: Assembler Services Guide*.)

As mentioned previously, LLA caches, in its address space, a copy of the directory entries for the libraries it manages. For modules that reside in LLA-managed libraries, the system can quickly search the directories in virtual storage instead of using I/O to search the directories on DASD.

## Planning To Use LLA

To use LLA, do the following:

1. Determine which libraries LLA is to manage
2. Code members of parmlib (see “Coding the Required Members of Parmlib” on page 1-36)
3. Control LLA through operator commands (see “Controlling LLA and VLF Through Operator Commands” on page 1-37).

When determining which data sets LLA is to manage, try to limit these choices to production load libraries that are rarely changed. Because LLA manages LNKLST libraries by default, you need only determine which non-LNKLST libraries LLA is to manage. If, for some reason, you do not want LLA to manage particular LNKLST libraries, you must explicitly remove such libraries from LLA management (as described in “Removing Libraries From LLA Management” on page 1-39).

### Using VLF With LLA

Because you obtain the most benefit from LLA when you have both LLA and VLF functioning, you should plan to use both. When used with VLF, LLA reduces the I/O required to fetch modules from DASD by causing selected modules to be staged in

VLF data spaces. LLA does not use VLF to manage library directories. When used without VLF, LLA eliminates only the I/O the system would use in searching library directories on DASD.

### LLA Notes

1. All LLA-managed libraries must be cataloged. This includes LNKLST libraries. A library must remain cataloged for the entire time it is managed by LLA. Please see “Recataloging LLA-Managed Data Sets While LLA is Active” on page 1-42 for additional information about recataloging LLA-managed libraries.
2. The benefits of LLA apply only to modules that are retrieved through the following macros: LINK, LINKX, LOAD, ATTACH, ATTACHX, XCTL and XCTLX.
3. LLA does not stage load modules in overlay format. LLA manages the directory entries of overlay format modules, but the modules themselves are provided through program fetch. If you want to make overlay format modules eligible for staging, you must re-linkedit the modules as non-overlay format. These reformatted modules might occupy more storage when they execute and, if LLA does not stage them, might take longer to be fetched.

## Coding the Required Members of Parmlib

LLA and VLF are associated with parmlib members, as follows:

- Use the CSVLLAxx member to identify the libraries that LLA is to manage.
- Use the COFVLFxx member to extend VLF services to LLA.

This section provides guidance information for coding the keywords of CSVLLAxx. For information about the required syntax and rules for coding CSVLLAxx and COFVLFxx, see *z/OS MVS Initialization and Tuning Reference*.

### Coding CSVLLAxx

Use CSVLLAxx to specify which libraries LLA is to manage and how it is to manage them.

IBM does not provide a default CSVLLAxx member (such as CSVLLA00). If you do not supply a CSVLLAxx member, LLA will, by default, manage only the libraries that are accessed through the LNKLST concatenation. If you supply a CSVLLAxx member, LLA manages the libraries you specify in CSVLLAxx (on the LIBRARIES keyword) in addition to any libraries that are accessed through the LNKLST concatenation. Because LLA manages the LNKLST libraries by default, it is not necessary to specify any LNKLST libraries on the LIBRARIES keyword.

***Using Multiple CSVLLAxx Members:*** To use more than one CSVLLAxx member at a time, specify the additional members to be used on the PARMLIB and SUFFIX keywords in the original CSVLLAxx member. The CSVLLAxx members pointed to by the PARMLIB and SUFFIX keywords must not point back to the original member, nor to each other.

You can use the PARMLIB and SUFFIX keywords to specify CSVLLAxx members that reside in data sets other than PARMLIB. You can then control LLA’s specifications without having update access to PARMLIB.

You can also use the PARMSUFFIX parameter of the CSVLLAxx parmlib member to specify additional CSVLLAxx members. PARMSUFFIX differs from the PARMLIB(dsn) SUFFIX(xx) statement in that no data set name is specified. PARMSUFFIX searches the logical parmlib for the CSVLLAxx member.

***Example of Multiple CSVLLAxx Members:*** Let's say you want two PARMLIBS (IMS.PARMLLA and AMVS.LLAPARMS) so that a TSO REXX exec can automatically activate a new module in LNKLST when it has copied the new module into a LNKLST library.

Do the following:

```
START LLA,LLA=IM,SUB=MSTR with CSVLLAIM as:

     FREEZE(-LNKLST-)
     PARMLIB(IMS.PARMLLA)
     SUFFIX(IA)
     PARMLIB(AMVS.LLAPARMS)
     SUFFIX(RX)
```

where:

```
AMVS.LLAPARMS (CSVLLARX) would contain the latest update
requested by the REXX exec, such as:

     REMOVE(...)/LIBRARIES(...)MEMBERS...
                    or
     PARMSUFFIX(...)
                    or
     PARMLIB(...) SUFFIX(...)
```

The REXX exec and either:

- use multiple members and use the PARMSUFFIX to identify them, or
- move the old CSVLLARX to CSVLLARn before building the new one.

***Storing Program Objects in PDSEs:*** In VMS/ESA 4.3 with DFSMS 1.1.0 and SMS active, you can produce a program object, and excutable program unit that can be stored in a partitioned data set extended (PDSE) program library. Program objects resemble load modules in function, but have fewer restrictions and are stored in PDSE libraries instead of PDS libraries. PDSE libraries that are to be managed by LLA must contain program objects. LLA manages both load and program libraries.

### Coding COFVLFxx

To have VLF stage load modules from LLA-managed libraries, you can use the default COFVLFxx member that is shipped in parmlib (COFVLF00), or, optionally, an installation-supplied COFVLFxx member. The installation-supplied member must contain a CLASS statement for LLA (see COFVLF00 for an example).

If you modify the COFVLFxx parmlib member, you must stop and restart VLF to make the changes effective.

If you want to use an installation-supplied COFVLFxx member instead of COFVLF00, do the following:

- Specify a CLASS statement for LLA in the alternative COFVLFxx member
- Specify the suffix of the alternative COFVLFxx member on the START VLF command. Otherwise, the system uses COFVLF00 by default.

For information about the required syntax and rules for coding the COFVLFxx member, see *z/OS MVS Initialization and Tuning Reference*.

## Controlling LLA and VLF Through Operator Commands

Use the following commands to control LLA:

- START LLA and START VLF

- STOP LLA and STOP VLF
- MODIFY LLA.

This section explains how to use commands to control LLA. For information about the required syntax and rules for entering commands, see *z/OS MVS System Commands*.

## Starting LLA

The START LLA command is included in the IBM-supplied IEACMD00 member of parmlib. Therefore, the system automatically starts LLA when it reads the IEACMD00 member during system initialization.

Although the system automatically starts LLA, it does not, by default, activate a CSVLLAxx member. To activate a CSVLLAxx member at system initialization, specify the suffix (xx) of the CSVLLAxx member in either of the following places:

- In the LLA procedure in SYS1.PROCLIB, as shown:

  ```
  //LLA PROC LLA=xx
  //LLA EXEC PGM=CSVLLCRE,PARM='LLA=&LLA'
  ```

  where 'xx' matches the suffix on the CSVLLAxx member to be used.
- On the START LLA command in the IEACMD00 member, as shown:

  ```
  COM='START LLA,SUB=MSTR,LLA=xx'
  ```

  where 'xx' matches the suffix on the CSVLLAxx member to be used.

If you do not supply a CSVLLAxx member, LLA will by default manage only the libraries that are accessed through the LNKLST concatenation.

When started, LLA manages both the libraries specified in the CSVLLAxx member as well as the libraries in the LNKLST concatenation.

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 the LLA procedure in PROCLIB.

***Starting LLA After System Initialization:*** IBM recommends that you specify SUB=MSTR on the START LLA command to prevent LLA from failing if JES fails. For example:

```
START LLA,SUB=MSTR,LLA=xx
```

where 'xx' matches the suffix on the CSVLLAxx member to be used, if any.

## Starting VLF

LLA provides better performance when VLF services are available, so it is better (although not necessary) to start VLF before LLA. However, the operation of LLA does not depend on VLF.

To allow VLF to be started through the START command, create a VLF procedure, or use the following procedure, which resides in SYS1.PROCLIB:

```
//VLF     PROC  NN=00
//VLF     EXEC  PGM=COFMINIT,PARM='NN=&NN'
//IEFPARM DD    DISP=SHR,DSN=SYS1.PARMLIB
```

When you issue the START VLF command, the VLF procedure activates the IBM-supplied COFVLF00 member, which contains a CLASS statement for LLA.

## Stopping LLA and VLF

You can stop LLA and VLF through STOP commands. Note that stopping VLF purges all the data in VLF data spaces for all classes defined to VLF (including LLA), and will slow performance.

If LLA or VLF is stopped (either by a STOP command or because of a system failure), you can use a START command to reactivate the function.

## Modifying LLA

You can use the MODIFY LLA command to change LLA dynamically, in either of the following ways:

- MODIFY LLA,REFRESH. Rebuilds LLA's directory for the entire set of libraries managed by LLA. This action is often called a *complete refresh* of LLA.
- MODIFY LLA,UPDATE=xx. Rebuilds LLA's directory only for specified libraries or modules. *xx* identifies the CSVLLAxx member that contains the names of the libraries for which directory information is to be refreshed. This action is often called a *selective refresh* of LLA. (For details, see "Identifying Members for Selective Refreshes.")

When an LLA-managed library is updated, the version of a module that is located by a directory entry saved in LLA will differ from the version located by the current directory entry on DASD for that module. If you update a load module in a library that LLA manages, it is a good idea to follow the update by issuing the appropriate form of the MODIFY LLA command to refresh LLA's cache with the latest version of the directory information from DASD. Otherwise, the system will continue to use an older version of a load module.

**Note:** Applications can use the LLACOPY macro to refresh LLA's directory information. For information about the LLACOPY macro, see *z/OS MVS Programming: Authorized Assembler Services Guide*.

***Identifying Members for Selective Refreshes:*** In CSVLLAxx, specify the MEMBERS keyword to identify members of LLA-managed libraries for which LLA-cached directory entries are to be refreshed during selective refreshes of LLA. If you issue the MODIFY LLA,UPDATE=xx command to select a CSVLLAxx member that has libraries specified on the MEMBERS keyword, LLA will update its directory for each of the members listed on the MEMBERS keyword.

Selectively refreshing 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.

For more information about specifying the MEMBERS keyword, see the description of the CSVLLAxx member in *z/OS MVS Initialization and Tuning Reference*.

## Removing Libraries From LLA Management

In CSVLLAxx, specify the REMOVE keyword for libraries that are to be removed dynamically from LLA management. If you issue the MODIFY LLA,UPDATE=xx command (selective refresh) to select a CSVLLAxx member that lists libraries on the REMOVE keyword, LLA no longer provides directory entries or staged modules for these libraries, regardless of whether the libraries are included in the LNKLST concatenation. (Note that you cannot use REMOVE to change the order or contents of the LNKLST concatenation itself.)

You can also use the MODIFY LLA,REFRESH command (complete refresh) to remove libraries from LLA management. This command rebuilds the entire LLA directory, rather than specified entries in LLA's directory. To limit the adverse effects on performance caused by an LLA refresh, whenever possible, use a selective refresh of LLA instead of a complete refresh, or stopping and restarting LLA.

In any case, when LLA directory entries are refreshed, LLA discards directory information of the associated module and causes VLF (if active) to remove the module from the VLF cache. This reduces LLA's performance benefit until LLA stages them again. Because LLA stages modules using the cached directory entries, you should refresh LLA whenever a change is made to an LLA-managed data set.

The MODIFY LLA command does not reload (or refresh) modules that are already loaded into virtual storage, such as modules in long-running or never-ending tasks.

For more information about specifying the REMOVE keyword, see the description of the CSVLLAxx member in *z/OS MVS Initialization and Tuning Reference*.

## Modifying Shared Data Sets

You can allow two or more systems to share access to the same library directory. When modifying or stopping LLA in this case, your changes must take effect simultaneously on all systems that share access to the directory.

In cases where you simply want to update an LLA-managed data set, it is easier to remove the data set from LLA management and update it, rather than stopping LLA on all systems. To do so, enter a MODIFY LLA,UPDATE=xx command on each system that shares access to the data set, where 'xx' identifies a CSVLLAxx member that specifies, on the REMOVE keyword, the data set to be removed from LLA management.

When you have completed updating the data set, enter the MODIFY LLA,UPDATE=xx command again, this time specifying a CSVLLAxx parmlib member in which the keyword LIBRARIES specifies the name of the data set.

In any case, whenever multiple systems share access to an LLA-managed data set, STOP, START, or MODIFY commands must be entered for LLA on all the systems.

## Using the FREEZE|NOFREEZE Option

For an LLA-managed library, use the FREEZE|NOFREEZE option to indicate whether the system is to search the LLA-cached or DASD copy of a library directory. With FREEZE|NOFREEZE, which you code in the CSVLLAxx member, you specify on a library-by-library basis which directory copy the system is to search, as follows:

- If you specify FREEZE, the system uses the copy of the directory that is maintained in the LLA address space (the library is "frozen").
- If you specify NOFREEZE, the system searches the directory that resides in DASD storage.

The system always treats libraries in the LNKLST concatenation as frozen. Therefore, you need only specify the FREEZE option for non-LNKLST libraries, or for LNKLST libraries that are referenced through TASKLIBs, STEPLIBs, or JOBLIBs.

When an LLA-managed library is frozen, the following is true:

- Users of the library always receive versions of the library's modules that are pointed to by the LLA-cached directory.
- Users do not see any updates to the library 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.
- If the version of a requested module matches the version of the module in VLF, the users receive the module from VLF. Otherwise, users receive the module from DASD.

To take full advantage of LLA's elimination of I/O for directory search, specify FREEZE for as many read-only or infrequently updated libraries as appropriate.

Having NOFREEZE in effect for an LLA-managed library means that your installation does not eliminate I/O while searching the library's directory. However, LLA can still improve performance when the system fetches load modules from the VLF data space instead of DASD storage.

Table 1-2 summarizes the affects of the FREEZE|NOFREEZE option on directory search and module fetch.

*Table 1-2. FREEZE|NOFREEZE Processing*

| **Action** | **FREEZE or NOFREEZE in Effect** | **LNKLST Libraries Accessed From LNKLST** | **LNKLST Libraries Accessed Outside LNKLST** | **Other Non-LNKLST Libraries** |
|---|---|---|---|---|
| Directory Search | *FREEZE* | LLA directory is used. | LLA directory is used. | LLA directory is used. |
| | *NOFREEZE* | LLA directory is used. | DASD directory is used (I/O occurs) | DASD directory is used (I/O occurs) |
| Module Fetch | *FREEZE* | Requestor receives LLA version of module (from VLF data space or DASD). | Requestor receives LLA version of module (from VLF data space or DASD). | Requestor receives LLA version of module (from VLF data space or DASD). |
| | *NOFREEZE* | Requestor receives LLA version of module (from VLF data space or DASD). | Requestor receives most current version of module (from DASD or VLF data space, if staged). | Requestor receives most current version of module (from DASD or VLF data space, if staged). |

You can change the FREEZE|NOFREEZE status of an LLA-managed library at any time through the MODIFY LLA command. Changing a library from NOFREEZE to FREEZE also causes a refresh of the directory information for that library (note that when a library is refreshed, all of its modules are destaged from the VLF data space, which will slow performance until new versions are staged).

For more information about specifying the FREEZE|NOFREEZE option, see the description of the CSVLLAxx member in *z/OS MVS Initialization and Tuning Reference*.

## Changing LLA-Managed Libraries

After changing a module in an LLA-managed library, IBM recommends that you refresh LLA for the library. A module that is changed and has a new location is not considered for staging until that member is refreshed.

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.

***LLA ENQ Consideration:*** By default, LLA allocates the libraries it manages as DISP=SHR. This means that if a job attempts to allocate an LLA-managed library as DISP=OLD, the job is enqueued until LLA is stopped or the library is removed from LLA management. Before adding a library to the libraries that LLA manages, review and, if necessary, change the jobs that reference the library.

***Using the GET_LIB_ENQ Keyword:*** The GET_LIB_ENQ keyword in the CSVLLAxx member allows you to prevent LLA from obtaining an exclusive enqueue on the libraries it manages. If you specify GET_LIB_ENQ (NO), your installation's jobs can update, move, or delete LLA-managed libraries while other users are accessing the libraries. GET_LIB_ENQ (NO) is generally not recommended, however, because of the risks it poses to data set integrity.

***Compressing LLA-Managed Libraries:*** If you compress an LLA-managed library, LLA continues to provide the obsolete directory entries. For members that have been staged to the VLF data space, the system will operate successfully. If the member is not currently staged, however, the cached obsolete directory entry can be used to fetch the member at the old TTR location from DASD.

Because using obsolete directory entries can cause such problems as abends, breaches of system integrity, and system failures, use the following procedure to compress LLA-managed libraries:

1. Issue a MODIFY LLA,UPDATE=xx command, where the CSVLLAxx parmlib member includes a REMOVE statement identifying the library that needs to be compressed.
2. Compress the library
3. Issue a MODIFY LLA,UPDATE=xx command, where the CSVLLAxx parmlib member includes a LIBRARIES statement to return the compressed library to LLA management.

This procedure causes all members of that library to be discarded from the VLF data space. The members are then eligible to be staged again.

## Recataloging LLA-Managed Data Sets While LLA is Active

LLA dynamically allocates the library data sets it manages. During unallocation (when LLA is stopped or the system is reIPLed), the system recatalogs the library data sets to the volume that was current at LLA initialization. Therefore, to recatalog an LLA-managed library data set while LLA is active, do the following:

1. Remove the library data set from LLA. (Issue a MODIFY LLA,UPDATE=xx command, where *xx* identifies the suffix of the CSVLLAxx parmlib member that includes a REMOVE statement that identifies the library data set to be removed from LLA management.)
2. Recatalog the library data set.

3. Return the library data set to LLA. (Issue a MODIFY LLA,UPDATE=xx command, where *xx* identifies the suffix of the CSVLLAxx parmlib member that includes a LIBRARIES statement that identifies the recataloged library data set to be returned to LLA management.) Recataloged LNKLST libraries cannot be put back into LLA management. This causes fetch failures.

## Allocation Considerations

Before a job can execute, the operating system must set aside the devices, and space on the devices, for the data that is to be read, merged, sorted, stored, punched, or printed. In MVS, the "setting-aside" process is called **allocation**.

MVS assigns **units** (devices), **volumes** (space for data sets), and **data sets** (space for collections of data) according to the *data definition* (DD) and *data control block* (DCB) information included in the JCL for the job step.

When the data definition or DCB information is in the form of text units, the allocation of resources is said to be *dynamic*. Dynamic allocation means you are requesting the system to allocate and/or deallocate resources for a job step while it is executing.For details on the use of dynamic allocation, see *z/OS MVS Programming: Authorized Assembler Services Guide*.

## Serialization of Resources During Allocation

When the system is setting aside non-sharable devices, volumes and data sets for a job or a step, it must prevent any other job from using those resources during the allocation process. To prevent a resource from changing status while it is being allocated to a job, the system uses **serialization**. Serialization during allocation causes jobs to wait for the resources and can have a major impact on system performance. Therefore, the system attempts to minimize the amount of time lost to serialization by providing a specific order of allocation processing. See Table 1-3.

Knowing the order in which the system chooses devices, you can improve system performance by making sure your installation's jobs request resources that require the least possible serialization.

The system processes resource allocation requests in the order shown in Table 1-3. As you move down the list, the degree of serialization – and processing time – increases.

*Table 1-3. Processing Order for Allocation Requests Requiring Serialization*

| KINDS OF ALLOCATION REQUESTS | SERIALIZATION REQUIRED |
|---|---|
| Requests requiring no specific units or volumes; for example, DUMMY, VIO, and subsystem data sets. | No serialization. |
| Requests for sharable units: DASD that have permanently resident or reserved volumes mounted on them. | No serialization. |
| Teleprocessing devices. | Serialized on the requested devices. |
| Pre-mounted volumes, and devices that do not need volumes. | Serialized on the group(s) of devices eligible to satisfy the request. A single generic device type is serialized at a time. |
| Online, nonallocated devices that need the operator to mount volumes. | Serialized on the group(s) of devices eligible to satisfy the request. A single generic device type is serialized at a time. |

*Table 1-3. Processing Order for Allocation Requests Requiring Serialization (continued)*

| KINDS OF ALLOCATION REQUESTS | SERIALIZATION REQUIRED |
|---|---|
| All other requests: offline devices, nonsharable devices already allocated to other jobs. | Serialized on one or more groups of devices eligible to satisfy the request. A single generic device type is serialized at a time. |

## Improving Allocation Performance

You can contribute to the efficiency of allocation processing throughout your installation in several ways:

- For **devices:**
  - Use the **device preference table**, specified through the Hardware Configuration Definition. (HCD) to set up the order for device allocation. See the *z/OS MVS Device Validation Support* appendix for a listing of the device preference table. installation's devices as *esoteric groups*, and to group them for selection by allocation processing.
- Use the **eligible device table** (EDT) to identify your

  For more information on the device preference table and the EDT, see *z/OS HCD Planning*.
- For **volumes**, use the **VATLSTxx** members of SYS1.PARMLIB to specify volume attributes at IPL.
- For **data sets**, you can specify the JCL used for your installation's applications according to the device selection criteria you have set up through the HCD process and IPL.

## The Volume Attribute List

In MVS, each online device in the installation has a **mount attribute** and each mounted volume has a **use attribute**.

The **mount** attributes determine when the volume on that device should be removed. This information is needed when selecting a device and during step unallocation processing. The **mount** attributes are:
- Permanently resident
- Reserved
- Removable.

The **use** attributes determine the type of nonspecific requests eligible to that volume. The **use** attributes are:
- Public
- Private
- Storage.

Allocation routines use the volumes' **mount** and **use** attributes in selecting devices to satisfy allocation requests. Thoughtful selection of the use and mount attributes is important to the efficiency of your installation. For example, during allocation, data sets on volumes marked permanently resident or reserved are selected first because they require no serialization, thus minimizing processing time.

During system initialization, you can assign volumeattributes to direct access volumes by means of the **volume attribute list**. The volume attribute list is defined at IPL time using the VATLSTxx member of SYS1.PARMLIB.

After an IPL, you can assign volume attributes to both direct access and tape volumes using the MOUNT command. The USE= parameter on the MOUNT

command defines the use attribute the volume is to have; a mount attribute of *reserved* is automatic. For the details of using the MOUNT command, see *z/OS MVS System Commands*.

When volumes are not specifically assigned a use attribute in the VATLSTxx member or in a MOUNT command, the system assigns a default. You can specify this default using the VATDEF statement in VATLSTxx. If you do not specify VATDEF, the system assigns a default of *public*. For details on including a volume attribute list in IEASYSxx,and on coding the VATLSTxx parmlib member itself, see *z/OS MVS Initialization and Tuning Reference*.

## Use and Mount Attributes

Every volume is assigned use and mount attributes through an entry in the VATLSTxx member at IPL, a MOUNT command, or by the system in response to a DD statement.

The relationships between use and mount attributes are complex, but logical. The kinds of devices available in an installation, the kinds of data sets that will reside on a volume, and the kinds of uses the data sets will be put to, all have a bearing on the attributes assigned to a volume. Generally, the operating system establishes and treats volume attributes as outlined below.

### Use attributes

- **Private** — meaning the volume can only be allocated when its volume serial number is explicitly or implicitly specified.
- **Public** — meaning the volume is eligible for allocation to a temporary data set, provided the request is not for a specific volume and PRIVATE has not been specified on the VOLUME parameter of the DD statement.

  Both tape and direct access volumes are given the public use attribute.

  A public volume may also be allocated when its volume serial number is specified on the request.
- **Storage** — meaning the volume is eligible for allocation to both temporary and non-temporary data sets, when no specific volume is requested and PRIVATE is not specified. Storage volumes usually contain non-temporary data sets, but temporary data sets that cannot be assigned to public volumes are also assigned to storage volumes.

### Mount attributes

- **Permanently resident** — meaning thevolume cannot be demounted. Only direct access volumes can be permanently resident. The following volumes are always permanently resident:
  - All volumes that cannot be physically demounted, such as drum storage and fixed disk volumes
  - The IPL volume
  - The volume containing the system data sets

  In the VATLSTxx member, you can assign a permanently-residentvolume any of the three use attributes. If you do not assign a use attribute to a permanently-resident volume, the default is public.
- **Reserved** — meaning the volume is to remain mounted until the operator issues an UNLOAD command.

  Both direct access and tape volumes can be reserved because of the MOUNT command; only DASD volumes can be reserved through the VATLSTxx member.

The reserved attribute is usually assigned to a volume that will be used by many jobs to avoid repeated mounting and demounting.

You can assign a reserved *direct access* volume any of the three use attributes, through the USE parameter of the MOUNT command or the VATLSTxx member, whichever is used to reserve the volume.

A reserved *tape* volume can only be assigned the use attributes of private or public.

- **Removable** — meaning that the volume is not permanently resident or reserved. A removable volume can be demounted either after the end of the job in which it is last used, or when the unit it is mounted on is needed for another volume.

  You can assign the use attributes of private or public to a removable *direct access* volume, depending on whether VOLUME=PRIVATE is coded on the DD statement: if this subparameter is coded, the use attribute is private; if not, it is public.

  You can assign the use attributes of private or public to a removable *tape* volume under one of the following conditions:

  – **Private**
    - The PRIVATE subparameter is coded on the DD statement.
    - The request is for a specific volume.
    - The data set is nontemporary (not a system-generated data set name, and a disposition other than DELETE).

    **Note:** The request must be for a **tape only** data set. If, for example, an esoteric group name includes both tape and direct access devices, a volume allocated to it will be assigned a use attribute of public.

  – **Public**
    - The PRIVATE subparameter is not coded on the DD statement.
    - A nonspecific volume request is being made.
    - The data set is temporary (a system-generated data set name, or a disposition of DELETE).

Table 1-4 summarizes the mount and use attributes and how they are related to allocation requests.

*Table 1-4. Summary of Mount and Use Attribute Combinations*

| Volume State | Temporary Data Set | Nontemporary data set | How Assigned | How Demounted |
|---|---|---|---|---|
| | Type of Volume Request | | | |
| Public and Permanently Resident[1] | Nonspecific or Specific | Specific | VATLSTxx entry or by default | Always mounted |
| Private and Permanently Resident[1] | Specific | Specific | VATLSTxx entry | Always mounted |
| Storage and Permanently Resident[1] | Nonspecific or Specific | Nonspecific or Specific | VATLSTxx entry | Always mounted |
| Public and Reserved (Tape and direct access) | NonSpecific or Specific | Specific | Direct access: VATLSTxx entry or MOUNT command Tape: MOUNT command | UNLOAD or VARY OFFLINE commands |

*Table 1-4. Summary of Mount and Use Attribute Combinations (continued)*

| Volume State | Temporary Data Set | Nontemporary data set | How Assigned | How Demounted |
|---|---|---|---|---|
| | Type of Volume Request | | | |
| Private and Reserved (Tape and direct access) | Specific | Specific | Direct access: VATLSTxx entry or MOUNT command Tape: MOUNT command | UNLOAD or VARY OFFLINE commands |
| Storage and Reserved[1] | Nonspecific or Specific | Nonspecific or Specific | VATLSTxx entry or MOUNT command | UNLOAD or VARY OFFLINE commands |
| Public and Removable (Tape and direct access) | Nonspecific or Specific | Specific | VOLUME=PRIVATE is not coded on the DD statement. (For tape, nonspecific volume request and a temporary data set also cause this assignment.) | When unit is required by another volume; or by UNLOAD or VARY OFFLINE commands. |
| Private and Removable (Tape and direct access) | Specific | Specific | VOLUME=PRIVATE is coded on the DD statement. (For tape, a specific volume request or a nontemporary data set also cause this assignment). | At job termination for direct access; at step termination or dynamic unallocation for tape (unless VOL=RETAIN or a disposition of PASS was specified); or when the unit is required by another volume. |
| [1]Direct access volumes only. | | | | |

## The Nonsharable Attribute

Some allocation requests imply the exclusive use of a direct access device while the volume is demounted and/or mounted. The system assigns the **non-sharable attribute** to volumes that might require demounting during step execution.

When a volume is thus made non-sharable, it cannot be assigned to any other data set until the non-sharable attribute is removed at the end of step execution.

The following types of requests cause the system to automatically assign the non-sharable attribute to a volume:

- A specific volume request that specifies more volumes than devices.
- A nonspecific request for a *private* volume that specifies more volumes than devices.
- A volume request that includes a request for unit affinity to a preceding DD statement, but does not specify the same volume for the data set. For more information, see the discussion of unit affinity in *z/OS MVS JCL Reference*.
- A request for deferred mounting of the volume on which a requested data set resides.

Except for one situation, the system will NOT assign the non-sharable attribute to a permanently-resident or reserved volume. The exception occurs when the allocation request is for more volumes than units, and one of the volumes is reserved. The reserved volume is to share a unit with one or more *removable* volumes, which precede it in the list of volume serial numbers.

For example:

```
DSN=BCA.ABC,VOL=SER=(A,B),UNIT=DISK

where volume A is removable and
      volume B is reserved
```

In this example, *both* volumes are assigned the non-sharable attribute; neither of them can be used in another job at the same time.

To avoid this situation, do one of the following:
- Specify the same number of volumes as units
- Specify parallel mounting
- Set the mount attribute of volume A as resident or reserved.

***System Action:*** Table 1-5 shows the system action for sharable and non-sharable requests.

*Table 1-5. Sharable and Nonsharable Volume Requests*

| The Request is: | The Volume is Allocated: | |
|---|---|---|
| | **Sharable** | **Nonsharable** |
| Sharable | allocate the volume | wait[1] |
| Nonsharable | wait[1] | wait[1] |

[1] The operator has the option of failing the request. The request will always fail if waiting is not allowed.

For more detailed information on how an application's JCL influences the processing of allocation requests, see *z/OS MVS JCL Reference*.

For details on how dynamic allocation affects the use attributes of the volumes in your installation, see *z/OS MVS Programming: Assembler Services Guide*.

# Chapter 2. Auxiliary Storage Management Initialization

This chapter gives guidelines for the effective use of page data sets. Additional information on this subject appears under PAGE, NONVIO, and PAGTOTL parameters of parmlib member IEASYSxx in *z/OS MVS Initialization and Tuning Reference*.

## Page Operations

The paging controllers of the auxiliary storage manager (ASM) attempt to maximize input/output (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 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. Multiple local page data sets are recommended. Although the system requires only one local page data set, performance improvement can be obtained when local page data sets are distributed across 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 permitted.

The general intent of the ASM algorithms for page data set selection and channel program construction is to:

- *Use all available local page data sets:* 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. For certain types of requests, 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. 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.

## Page Data Set Sizes

Page data set sizes can affect system performance. The maximum number of slots for a page data set is 1,048,576 slots per data set. Note the following recommendations:

- *PLPA data set.* 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 twenty megabytes for common, as spilling will occur if the PLPA data set becomes full.

RMF reports can be used to determine the size requirements for these data sets. During system initialization, check the size of the page data sets in the page data set activity report. If the slots used values are very close to the slots allocated value, the size should be enlarged. For more information, see the description of the page data set activity report in *z/OS RMF Report Analysis*.

- *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 twenty megabytes.
- *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. Plan for local page data sets to be at no more than 30% of their capacity under normal system workloads. When utilization exceeds 30%, the slot allocation algorithms become less efficient, and may degrade I/O performance. All local page data sets should be about the same size, in number of slots. When they are not, the utilization of the smaller data sets will exceed the recommended levels before the larger data sets, causing performance to degrade.

## Page Data Set Protection

The page data set protection feature was introduced in z/OS V1R3 to help guard you from unintentionally IPLing with a page data set that is already in use. It does this by formatting and maintaining a status information record at the beginning of each page data set and by using an ENQ to serialize usage of the data sets.

The page data set protection feature prevents two systems from accidentally using the same physical data set. However, it is not possible to prevent the same data sets from being used when:

- The request to use the data set comes from a system outside of the GRS Ring/Star configuration.
- The installation has excluded the data sets from multi-system serialization. See *z/OS MVS Planning: Global Resource Serialization* for more information on SYSTEMS Exclusion RNL.

Page data sets are protected by a two-tier mechanism using:

- "SYSTEMS level ENQ".
- "Status Information Record" on page 2-3.

## SYSTEMS level ENQ

Page data sets are protected with a SYSTEMS level ENQ that contains the name of the page data set and the volser it resides on. The qname used on the ENQ is SYSZILRD, and the rname used on the ENQ is the dsname + volser. This ENQ is obtained during master scheduler initialization for the IPL-specified page data sets. For example, the ENQ would be obtained from the definitions in the IEASYSxx parmlib member. The ENQ is also obtained whenever a page data set is added or replaced after IPL.

A warning-level message is issued if the ENQ cannot be obtained successfully during IPL for the IPL-specified page data sets. Processing for the command is terminated if the ENQ cannot be obtained for a page data set that is being added or replaced via the PAGEADD or PAGEDEL command.

## Status Information Record

A data set status information record is written to every page data set. The status record identifies the system that uses the data set.

The status record is validated during IPL. If the record indicates that the data set is in use by another system, message ILR030A is issued and the system waits for an operator response to allow or disallow use of the data set.

Note that this feature does not offer full protection in the case of a page data set that was defined on, or was last used by a pre-z/OS V1R3 environment. This is because the status record used to perform this check did not exist prior to V1R3. The z/OS1.3 system will format the status record, issue ILR029I as an informational message and continue to use the data set (along with the other system).

Once the IPL is complete, the status record is validated on a regular interval. If concurrent use of a data set is detected, both systems will be terminated with a 025 wait state code. Catalog information will also be validated with the status record. If the data set is deleted or key catalog information changes, the system will be terminated with a 025 wait state code.

# Space Calculation Examples

Table 2-1 shows the values for page 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 data sets. RMF reports provide the minimum, maximum, and average number of slots in use for page data sets. Thus, you can use the reports to adjust data set sizes, as needed.

*Table 2-1. Page data set Values*

| Device Type | Slots/Cyl | Cyl/Meg |
|---|---|---|
| 3380 | 150 | 1.7 |
| 3390 | 180 | 1.42 |
| 9345 | 150 | 1.7 |

## Example 1: Sizing the PLPA Page Data Set, Size of the PLPA and Extended 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, 7 cylinders on a 3380 are needed for the 4-megabyte PLPA page data set. For the 3390, 6 cylinders are needed for the 4-megabyte PLPA page data set.

## Example 2: Sizing the PLPA Page Data Set, Size of the PLPA and Extended 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, 18 cylinders on a 3380 are needed for the 10.5-megabyte PLPA page data set. For the 3390, 15 cylinders are needed for the 10.5-megabyte PLPA page data set.

## Example 3: Sizing the Common Page Data Set

Use the size of the virtual common service area (CSA) and extended CSA, defined by the CSA= parameter in the IEASYSxx parmlib member, as the minimum size for the common page data set. If the system is IPLed with a specification of CSA=(3000,80000), then the total CSA specified is 83000 kilobytes (approximately 81 megabytes). However, CSA and extended CSA always end on a segment boundary, so the system may round the size up by as much as 1023 kilobytes each. That rounding could make the CSA size as large as 83 megabytes with the CSA=(3000,80000) specification. 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, 118 cylinders on a 3390 are needed to start with an 83-megabyte common page data set.

## Example 4: Sizing Local Page Data Sets

Assume that the master scheduler address space and Job Entry Subsystem (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 Time Sharing Option Extensions (TSO/E), multiply the maximum number of TSO/E 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 that is 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/E address spaces (50 TSO/E 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 (approximately 5%) buffer = 305 megabytes
```

*Therefore:* From the tables, 549 cylinders on 3380 type devices are necessary. For the 3390, 434 cylinders are necessary.

**Note:** Even when the local page data sets will fit on one 3390, 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. Users of data-in virtual may use sizable amounts of virtual storage which may put additional requirements on paging data sets.

The calculations shown here will provide enough local page space for the system to run. However, if a program continually requests virtual storage until the available local page data set space is constrained, this will not be enough space to prevent an auxiliary storage shortage.

Auxiliary storage shortages can cause severe performance degradation. If all local page data set space is exhausted, the system might fail. To avoid this, you should do one or more of the following:

- Use the IEFUSI user exit to set REGION limits for most address spaces and data spaces. If you have sufficient DASD, add enough extra local page data set space to accommodate one address space or data space of the size you allow in addition to the local page data set space the system usually requires, multiply the sum by 1.43, and allocate that amount of local page data set space. For more information about IEFUSI, see *z/OS MVS Installation Exits*.
- Over allocate local page data set space by 300% if you have sufficient DASD. This is enough auxiliary storage to prevent the system from reaching the auxiliary storage shortage threshold when the maximum amount of storage is obtained in a single address space or data space.

## 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 device. 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 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 causes the loss of all gains in performance resulting from ASM using the suspend or resume 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. 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/E sessions. VIO data set paging can be controlled by restricting it to certain page data sets through the use of directed VIO.

   VIO data set pages can be purged by a reIPL specifying CVIO (or create link pack area (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.

3. Use more than one local page data set, each on a unique device, even if the total required space is containable 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.

4. 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 the control unit is busy.

5. 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.

6. 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.

7. Do not share volumes that contain page data sets among multiple systems.

   **Reason:** While page data sets may be defined on volumes that contain shared non-paging data sets, they cannot be shared between systems.

8. Take one of the actions below to control the risk of auxiliary storage shortages. Auxiliary storage shortages have severe effects on system performance when they occur, and can also cause the system to fail. When a shortage occurs, the system rejects LOGON, MOUNT, and START commands and keeps address spaces with rapidly increasing auxiliary storage requirements from running until the shortage is relieved.
   a. Use the System Management Facility (SMF) Step Initiation Exit, IEFUSI, to limit the sizes of most address spaces and data spaces.
   b. If you do not establish limits using IEFUSI, consider over allocating local page space by an amount sufficient to allow a single address space or data space to reach the virtual storage limit (as might happen if a program looped obtaining storage) without exhausting virtual storage shortage.

   See “Example 4: Sizing Local Page Data Sets” on page 2-4 for more information about calculating local page data set space requirements.

## 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).
2. The space needed for areas of virtual storage: 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 estimate 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 must use the DEFINE PAGESPACE command of Access Method Services to pre-format and catalog each new page data set. To add the page data set, you can use the PAGEADD operator command, or specify the data set at the next IPL on the PAGE parameter.

**For More Information...**

- See *z/OS DFSMS Access Method Services for Catalogs* for information about the DEFINE PAGESPACE command, and on the related commands - ALTER and DELETE - used for the handling of VSAM data sets.
- See *z/OS MVS System Commands* for a description of the PAGEADD command.
- See the description of the PAGE parameter in parmlib member IEASYSxx in *z/OS MVS Initialization and Tuning Reference*.

## Deleting, Replacing or Draining Page Data Sets

You might need to remove a local page data set from the system for any of the following reasons:

- The hardware is being reconfigured.
- The hardware is generating I/O errors.
- The configuration of the page data set is being changed.
- System tuning requires the change.

The PAGEDEL command allows you to delete, replace or drain local page data set without an IPL. See *z/OS MVS 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 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/E 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 reIPL to specify an additional

pre-formatted and cataloged page data set. (See the description of the PAGE parameter of the IEASYSxx member in *z/OS MVS Initialization and Tuning Reference*.)

**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 *z/OS MVS Initialization and Tuning Reference* and the PAGEADD command in *z/OS MVS System Commands*).

**Q:** *Can we remove paging space from system use?*

**A:** Yes. Use the PAGEDEL command for local page 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:** *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 available page data sets. ASM selects these data sets in a circular order within each type of data set, subject to the availability of free space and the device response time.

**Q:** *How will expanded storage affect my paging configuration?*

**A:** Expanded storage helps reduce the paging load to auxiliary storage devices. Therefore, you might be able to reduce the paging configuration below what was previously required when no expanded storage was configured.

**Q:** *How will z/Architecture affect my paging configuration?*

**A:** Expanded storage is not supported in z/Architecture. You can still use local page data sets in z/Architecture; however, because you can have more real storage in z/Architecture, less paging is likely to be needed.

**Q:** *Will data-in-virtual users increase the need for paging data sets?*

**A:** Data-in-virtual does provide applications with functions that would encourage extensive use of virtual storage. Depending on the extent of the usage of data-in-virtual, paging data set requirements may increase.

# Chapter 3. The System Resources Manager

**Important**

Beginning with z/OS V1R3, WLM compatibility mode is no longer available. Accordingly, you can no longer use most of the functions described in this chapter. The IEAICSxx member, the IEAIPSxx member, and most options in the IEAOPTxx member (those options that previously worked in compatibility mode only) are no longer valid. The information has been left here for reference purposes, and for use on backlevel systems.

See *z/OS MVS Planning: Workload Management* for more information on WLM goal mode.

To a large degree, an installation's control over the performance of the system is exercised through the system resources manager (SRM).

"Section 1: Description of the System Resources Manager (SRM)" on page 3-2 discusses the types of control available through SRM, the functions used to implement these controls, and the concepts inherent in the use of SRM parameters. The parameters themselves are described in *z/OS MVS Initialization and Tuning Reference*.

"Section 2: Basic SRM Parameter Concepts" on page 3-14 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 *z/OS MVS Initialization and Tuning Reference* and should be referred to when necessary.

"Section 3: Advanced SRM Parameter Concepts" on page 3-41 discusses some more advanced topics that can be read after you understand the concepts in this section.

"Section 4: Guidelines and Examples" on page 3-51 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. Guidelines are presented for defining these parameters and several complete sample specifications are described. A list of potential problems along with guidelines for evaluation and adjustment of parameters are provided.

"Section 5: Installation Management Controls" on page 3-79 contains information about JCL statements and operator commands for SRM-related functions.

## 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 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 request? 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/E 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 4: Guidelines and Examples” on page 3-51.

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.

## Controlling SRM

You have two options for controlling SRM: through the IEAIPSxx and IEAICSxx parmlib members, or through the workload manager. Controlling SRM through parmlib members is called workload management compatibility mode, and controlling SRM through the workload manager is called goal mode. Some parameters in the IEAOPTxx parmlib member apply to both goal and compatibility mode.

Because controlling SRM in compatibility mode can be a complex process, we recommend that you use workload manager goal mode. With workload manager, you specify performance goals for work, and SRM adapts the system resources to meet the goals. SRM uses the same controls that exist today, but it sets them all dynamically based on the goals. For information on how to use workload manager, see *z/OS MVS Planning: Workload Management*.

While most information about how to use workload manager is in *z/OS MVS Planning: Workload Management*, many of the concepts that SRM uses dynamically

in goal mode are explained in this publication. This publication, however, explains all of the concepts for compatibility mode. The section describing the IEAOPTxx parameters explains which parameters are applicable in goal mode.

## 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 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 (ICS), 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/E, 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 SYSEVENT macro.

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 ICS 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 three distinct types of control to an installation:

- Domain control, for swappable address spaces
- Period control, for all address spaces
- Dispatching control, for all address spaces.

The installation control specification, IPS, and OPT contain the parameters used to modify these controls. These are the parameters you use for compatibility mode. The syntax and parameters of the IEAICSxx, IEAIPSxx, and IEAOPTxx parmlib members are described in *z/OS MVS Initialization and Tuning Reference*.

For goal mode, SRM still uses domain and dispatching controls, but it sets the values dynamically based on the performance goals for work defined in a service policy.

The remainder of this section describes the types of controls, 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/E commands and long-running TSO/E 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: Basic SRM Parameter Concepts" on page 3-14.

## Period Control

Additional flexibility of control is gained by dividing the life span of a transaction (a unit of work that is consuming service) 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/E 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 how long a transaction runs.

### Dispatching Control

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.

The installation sets dispatching priorities through the IPS, and can alter them as the address space's execution characteristics change.

## Functions

This topic discusses the functions used by SRM to implement the controls described in the previous section.

The functions are as follows:

1. Swapping (see "Swapping")
2. Dispatching of work (see "Dispatching of Work" on page 3-7)
3. Resource use functions (see "Resource Use Functions" on page 3-8)
4. Enqueue delay minimization (see "Enqueue Delay Minimization" on page 3-10)
5. I/O priority queueing (see "I/O Priority Queueing" on page 3-10)
6. Prevention of storage shortages (see "Prevention of Storage Shortages" on page 3-11)
7. Pageable frame stealing (see "Pageable Frame Stealing" on page 3-13).

### 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 through 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/E transactions.

There are several reasons for swapping. Some swaps are used for control of domains and the competition for resources between individual address spaces within a domain, while others provide control over system-wide performance and help increase the throughput.

***Domain-Related Swaps:***

- *Unilateral swap in:* If the number of a domain's address spaces that are in the multi-programming set less than the number the installation specified, or less than the number SRM has set for the swap-in target, SRM will swap in additional address spaces for that domain, if possible. See "Domains" on page 3-18 for information about swap-in targets.
- *Unilateral swap out:* If the number of a domain's address spaces that are in the multi-programming set is greater than the number the installation specified, or greater than the number SRM has set for the swap-out target, SRM will swap out address spaces from that domain. See "Domains" on page 3-18 for information about swap-out targets.
- *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 the multi-programming set (MPS)[1] 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 the multi-programming set is swapped out and the other address space is swapped in. This competition between address spaces is described in detail in “Section 2: Basic SRM Parameter Concepts” on page 3-14.

***System-Related Swaps:***

- *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.
- *Swaps to improve central storage usage:* The system will swap out an address space when the system determines that the current mix of address spaces is not best utilizing central storage. The system swaps out address spaces to create a positive effect on system paging and swap costs.
- *Swaps to improve system paging rate:* The system will swap out an address space when the system page fault rate exceeds the high threshold specified on the RCCPTRT parameter of the IEAOPTxx parmlib member that was out too long:
- *Swap out an address space to make room for an address space:* The system will swap in an address space when the system determines that it has been out longer than its recommendation value would dictate. See “Working Set Management” on page 3-9 for information about the recommendation value.
- *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 central storage that are not executable for a fixed interval will be swapped. (See “Logical Swapping” on page 3-8.)
- *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.
- *Transition Swap:* A transition swap occurs when the status of an address space changes from swappable to nonswappable. 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.

## Swap Recommendation Value

SRM calculates a swap recommendation value to determine which address spaces to use in an exchange or unilateral swap. A high swap recommendation value

1. The multi-programming set consists of those address spaces that are in central storage and are eligible for access to the processor.

indicates that the address space is more likely to be swapped in. As the swap recommendation value decreases, that address space is more likely to be swapped out.

The swap recommendation value for an address space that is swapped in ranges from 100 to -999 as service is accumulated. An address space must accumulate enough CPU service to justify the cost of a swap out. Once the swap recommendation value goes below 0, the address space is ready to be swapped out in an exchange swap.

The swap recommendation value for an address space that is swapped out and ready to come in ranges from 0 to 998 as the address space remains out. Once the swap recommendation value goes above 100, the address space has been out long enough to justify the cost of the exchange swap.

For an address space that is swapped out but not ready to come in, the swap recommendation value as reported by the RMF Monitor II ASD report is meaningless. The swap recommendation value for an address space that is out too long is reported as 999. Also, if an address space has been assigned long-term storage protection (as described in the "Storage Protection" section of the "Workload Management Participants" chapter in *z/OS MVS Planning: Workload Management*), then the swap recommendation value is 999.

If exchange swap frequency results in too much paging or infrequent access to the multi-programming set for important workloads, use the SWAPRSF parameter in the IEAOPTxx parmlib member. The swap recommendation value reflects how close the address space is to accumulating enough service to account for the overhead of swapping the address space out. The swap recommendation value of an address space that is out will reflect how close the address space is to having stayed out long enough to justify the overhead of swapping the address space in. Setting a low SWAPRSF value will cause the swap recommendation value to change more rapidly and SRM to initiate an exchange swap more readily.

For monitored address spaces, SRM calculates a working set manager recommendation value that can override the swap recommendation value. See "Working Set Management" on page 3-9 for information about the working set manager recommendation value.

## Dispatching of Work

Dispatching of work is done on a priority basis. That is, the ready work with the highest priority is dispatched first. The total range of priorities is from 0 to 255. This range is divided into 16 sets of 16 priorities each.

**Note:** Certain system address spaces execute at the highest priority and are exempt from installation prioritization decisions. Within each of the 16 sets of priorities, 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. These waits include system imposed waits (for example, page faults from DASD). Those address spaces that are considered to be CPU-bound are assigned lower dispatching

priorities, within this group, than those that 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.

### 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.

## 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 - The installation can influence 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.
- Logical swapping - SRM automatically performs logical swapping when sufficient central storage is available. The installation can influence or eliminate logical swapping via threshold values set in the OPT.
- Working set management - SRM automatically determines the best mix of work in the multiprogramming set (MPS) and the most productive amount of central storage to allocate to each address space.

### Multiprogramming Level Adjusting

SRM monitors system-wide utilization of resources, such as the CPU and paging subsystem, and seeks to alleviate imbalances, that is, over-utilization or underutilization. This is accomplished by periodically adjusting the number of address spaces that are allowed in central storage and ready to be dispatched for appropriate domains (multi-programming set).

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 access to the processor for that domain, thereby increasing utilization of the system.

### Logical Swapping

In order to use central storage more effectively and reduce processor and channel subsystem overhead, the SRM logical swap function attempts to prevent the automatic physical swapping of address spaces. Unlike a physically-swapped address space, where the LSQA, fixed frames, and recently-referenced frames are placed in expanded storage or on auxiliary storage, SRM keeps the frames that belong to a logically-swapped address space in central storage.

Address spaces swapped for wait states (for example, TSO/E terminal waits) are eligible to be logically swapped out whenever the think time associated with the address space is less than the system threshold value. SRM adjusts this threshold value according to the demand for central storage. SRM uses the unreferenced interval count (UIC) to measure this demand for central storage. As the demand for central storage increases, SRM reduces the system threshold value; as the demand decreases, SRM increases the system threshold value.

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; setting these threshold values affects how SRM measures the demand for central storage.

SRM logically swaps address spaces if the real frames they own are not required to immediately replenish the supply of available real frames. Any address space subject to a swap out can become logically swapped as long as there is enough room in the system. Address spaces with pending transition swaps, request swaps, or those marked as swaps due to storage shortages are the only address spaces that are physically swapped immediately.

Large address spaces that have been selected to be swapped in order to replenish central storage are trimmed before they are swapped. The trimming is done in stages and only to the degree necessary for the address space to be swapped. In some cases, it may be necessary to trim pages that have been recently referenced in order to reduce the address space to a swappable size. SRM considers a swappable size to be 2 megabytes. However, you can override this value by indicating a working set size (MCCMAXSW) greater than 2 megabytes in the IEAOPTxx parmlib member.

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 only when it is necessary to replenish the supply of available frames.

## Working Set Management

SRM automatically determines the best mix of work in the multiprogramming set (MPS) and the most productive amount of central storage to allocate to each address space within MPL constraints.

In order to achieve this, SRM monitors the system paging, page movement, and swapping rates and productive CPU service for all address spaces (except, when GOAL mode is off, those address spaces that are either storage isolated or non-swappable) to detect when the system may run more efficiently with selected address space working sets managed individually. If the system is spending a significant amount of resources for paging, SRM will start monitoring the central storage of selected address spaces.

Once SRM decides that an address space should be monitored, SRM activates implicit block paging from expanded storage and collects additional address space data. Based on this data, SRM may discontinue implicit block paging.

For monitored address spaces, SRM calculates a working set manager recommendation value that can override the swap recommendation value. See "Swap Recommendation Value" on page 3-6 for information about the swap recommendation value. The working set manager recommendation value measures the value of adding the address space to the current mix of work in the system. Even when the swap recommendation value indicates that a specific address space should be swapped in next, the working set manager recommendation value might indicate that the address space should be bypassed. In order to ensure that no address space is repeatedly bypassed, the system swaps in a TSO/E user 30 seconds after being bypassed. For all other types of address spaces, the system will swap in the address space 10 minutes after being bypassed.

If a monitored address space is paging heavily, SRM may manage its central storage usage. When an address space is managed, SRM imposes a central storage target (implicit dynamic central storage isolation maximum working set) on an address space.

## 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 central 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: Basic SRM Parameter Concepts” on page 3-14.) 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.

## DASD Device Allocation

Device allocation selects the most responsive DASD devices as candidates for permanent datasets on mountable devices (JCL specifies nonspecific VOLUME information or a specific volume and the volume is not mounted).

The ability of SRM to control DASD device allocation is limited by the decision an installation makes at system installation 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 DASD devices that are equally acceptable for scheduler allocation. This set of devices does not necessarily include all the DASD devices placed in an esoteric group during system installation. At that time, an esoteric group is defined by the UNITNAME macro and entered in the eligible device table (EDT). During system installation each esoteric group is partitioned into subgroups if either of the following conditions occurs:

- The group includes DASD devices that are common with another esoteric group.

- The group includes DASD devices that have certain generic differences. System installation partitions only esoteric groups that consist of magnetic tape or direct access devices.

For example, assume that you specify the following at system installation time:

```
UNITNAME=DASD,UNIT=((470,7),(478,8),(580,6))
UNITNAME=SYSDA,UNIT=((580,6))
```

Because of the intersection with SYSDA (580,6), the DASD 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 DASD devices from the next subgroup. Using the previous example, when a job requests UNIT=DASD, 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 DASD devices in the first group have been allocated, allocation tells SRM to select devices from the second group (580-585).

## 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
- SQA
- Pageable frames.

### 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 target MPL (both in and out targets) in each domain is set to its minimum value.
- The operator is informed of the shortage.
- Choosing from a subset of swappable address spaces, SRM stops the address space(s) acquiring slots at the fastest rate and prepares the address space for swap-out (logical swap). 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.

If the percentage of auxiliary slots allocated continues to increase (constant MCCASMT2), SRM informs the operator that a critical shortage exists. SRM then 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.

When the shortage has been alleviated, the operator is informed and SRM halts its efforts to reduce the demand for auxiliary storage.

### 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.

## Pageable Frames

SRM attempts to ensure that enough pageable central 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.

When SRM detects a shortage of pageable frames caused by an excess of fixed or DREF storage, SRM uses event code ENVPC055 to signal an ENF event. This occurs in both goal mode and compatibility mode. When the shortage is relieved, SRM signals another ENVPC055 event to notify listeners that the shortage is relieved. SRM will not raise the signal for the “shortage relieved” condition until a delay of 30 seconds following the most recent occurrence of a fixed storage shortage. The intent of signalling this event is to give system components and subsystems that use fixed or DREF storage an opportunity to help relieve the shortage.

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.

The following system conditions can cause a shortage of pageable storage:

- Too many address spaces are already in storage.

  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 “OPT Concepts” on page 3-40 for information on MPL load adjustment and logical swapping.)
- Too much page fixing is taking place. One or more address spaces are using substantial amounts of storage either through explicit requests to fixed virtual storage or by obtaining virtual storage that is page fixed by attributes such as LSQA or SQA.

There are different types of pageable storage shortages:

- A shortage of pageable storage below 16 megabytes.
- A shortage when pageable storage has reached a threshold.
- A shortage when fixed, DREF, and expanded storage allocated to CASTOUT=NO ESO hiperspaces has reached a threshold.

The thresholds that SRM uses can be modified by the MCCFXEPR and MCCFXTPR 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 physically 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 indices 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. See “Domain Importance” on page 3-21 for more information on contention indices.

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.
3. Prevents any unilateral swap-in, except for domain zero.

When the shortage of pageable storage is relieved, SRM waits for a delay of 30 seconds following the most recent occurrence of a fixed shortage, and then:

- 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.

## Pageable Frame Stealing

Pageable frame stealing is the process of taking an assigned central storage 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 there is a demand for pageable frames, 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 SRM and RSM. Each address space is examined, along with the common service area (CSA) and the pageable-link pack area (PLPA). 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.

Stealing takes place strictly on a demand basis, that is, there is no periodic stealing of long-unreferenced frames.

For systems with no expanded storage, 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. For systems with expanded storage, the storage isolation function may not affect page stealing but may affect expanded storage migration the same way it affects page stealing in systems with no expanded storage.

SRM modifies the stealing process for address spaces that it is managing. For a managed address space, SRM attempts to enforce the address space's central storage target that was set when SRM decided that the address space was to be managed.

## Section 2: Basic 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 *z/OS MVS Initialization and Tuning Reference* and should be referred to when necessary. *z/OS MVS Initialization and Tuning Reference* also explains which OPT parameters are applicable in workload management goal mode.

The topics are presented in the following order:

- IPS concepts:
  - Performance group
  - Service
  - Service definition coefficients
  - Performance period
  - Domains
  - Service rates
  - Domain importance
  - Dispatching priorities
  - TSO/E response time control
  - IPS Example
  - SET IPS operator command.
- Installation control specification concepts:
  - Subsystem sections
  - Transaction entries
  - Assignment of control performance groups
  - Assignment of report performance groups
  - Substring notation
  - SET ICS operator command
  - Examples.
- OPT concepts:
  - MPL adjustment control
  - Other options.

Besides 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 "IPS Example" on page 3-23, a complete IPS has evolved. At that point, the syntax is used for the first time, requiring the reader to refer to *z/OS MVS Initialization and Tuning Reference*.

The discussion of installation control specification and OPT concepts is also dependent on preceding topics, that is, on IPS parameter concepts.

## IPS Concepts

The parameters discussed in this part are specified in the IEAIPSxx member of SYS1.PARMLIB (for short, called the IPS).

### 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 through the installation control specification (the IEAICSxx member of SYS1.PARMLIB). If an installation control specification is not in effect, batch job steps and TSO/E 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" on page 3-24.)

### Service

One of the basic functions of SRM is to monitor the dynamic performance characteristics of all address spaces under its control 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 transactionis a way to delineate a unit of work that is consuming service. For batch jobs, a transaction corresponds to a job or job step scheduled by JES. For APPC/MVS transaction programs, a transaction corresponds to work scheduled by the IBM-supplied ASCH transaction scheduler. For TSO/E work, a transaction normally corresponds to a command or terminal interaction.

**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.

Since only one transaction can be active in an address space at one time, and it is in fact 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.

A new transaction is defined for an active TSO/E user whenever any of the following occur:

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/E internal stack (as with a command in a CLIST).

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)
+ (SRB x SRB Service Units)
+ (IOC x I/O Service Units)
+ (MSO x Storage Service Units)

where CPU, IOC, MSO, and SRB are installation defined *service definition coefficients* and:

**CPU Service Units =**
task (TCB) execution time, multiplied 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.

**SRB Service Units =**
service request block (SRB) executiontime for both local and global SRBs, multiplied 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.

**I/O Service Units =**
measurement of individualdataset I/O activity and JES spool reads and writes for all datasets 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 datasets 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 =**
(central page frames) x (CPU serviceunits) 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 central storage 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.

## 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

## 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 theirexecution characteristics change. For example, if a domain includes a variety of TSO/E users, transactions with greatly differing life spans will be competing with one another. If most 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/E workload 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/E short domain, dispatching priority = a high fixed priority

**Set 2:** TSO/E medium domain, dispatching priority = a moderate fixed priority

**Set 3:** TSO/E long domain, dispatching priority = a low mean-time-to-wait priority

Assume also that three performance periods have been defined, so 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/E 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 through a unique identifier, the *performance group number (PGN)*.

## Domains

Domains, as discussed in “Domain Control” on page 3-4, 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 achieved by specifying the number of address spaces, in a domain, that can be in central storage and allowed access to the processor, at a given time. 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 *in-target MPL* and an *out-target MPL* where each target is adjustable up or down in response to changing activity levels.

The in-target MPL is the basis for SRM domain control since it represents the minimum number of a domain’s address spaces that are allowed in central storage and allowed access to the processors at any one time. When the system is not constrained, SRM will allow address spaces into the domain up to the out-target level. This allows for address spaces that are transitioning into a new domain because of period-to-period migration to continue to run, rather than get swapped out. Also, if storage is available, additional address spaces can get swapped into the domain up to the out-target.

The range of the in-target MPL and out-target MPL is subject to the installation-specified constraints (minMPL and maxMPL on the CNSTR parameter in the IPS), that is, the in-target MPL and the out-target are always between minMPL and maxMPL. The installation, therefore, is able 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. In this case, SRM will not create a range.

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 *z/OS MVS Initialization and Tuning Reference* for value ranges).

***Domain Examples:*** Some basic examples will now be discussed to show 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 allows 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 dispatching 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/E commands. Placing such work into a unique domain with minMPL equal to the expected maximum number of ready users will ensure that each address space is swapped in when 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 the multi-programming set.

*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 the multi-programming set at one time.

For example, TSO/E users doing resource intensive processing from the terminal can slow down the system considerably. Placing such users into a domain with minMPL = 1 and maxMPL = 1 would have the effect of serializing such work, since only one address space of this type would be allowed in the multi-programming set 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 the multi-programming set to two, even if, for example, eight initiators were started.

## 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:

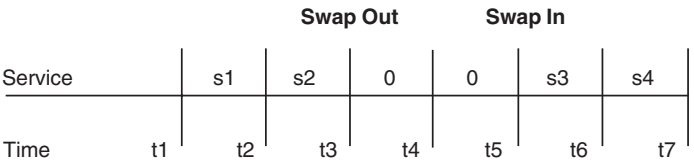
| | |
|---|---|
| Transaction Resident Time | - swapped in time |
| Transaction Out Time | - swapped out (ready) time |
| Transaction Long Wait Time | - swapped out (not ready) time |
| Transaction Active Time | - Transaction Resident Time + Transaction Out Time |
| Transaction Elapsed Time | - Transaction 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 called *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.

***Determining the Service Rate for a Domain:*** To determine the desired service rate for a domain, begin with the desired response time. If you want first period TSO/E work to complete in .1 — .2 seconds, consider the following:

$$\text{service rate} = \frac{\text{duration}}{\text{response time}}$$

For example, for a duration of 200 service units, specifying a response time of .1 — .2 seconds would indicate that you needed to deliver a service rate of 1000 — 2000 service units/second per user. This is an average per user. Therefore, code ASRV=(1000,2000).

For DSRV, multiply the number of address spaces allowed in the domain by the service rate.

## 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 is able to influence SRM's decision about which domain's target MPL will be increased or decreased by assigning to each domain an importance. This is determined by the IPS by the CNSTR keyword by specifying minMPL and maxMPL and by specifying one of the following for the domain:

- Total service
- Average service
- Fixed contention index.

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 under-used and seeks to raise some domain's target MPL to allow more address spaces in the multi-programming set.

SRM's decision is based on each domain's contention index. You can either specify a constant, or *fixed*,contention index for a domain, or you can specify a range of service values that cause the system to determine the contention index. By specifying a fixed contention index (through the FIXCIDX keyword in the IPS), you set the domain's importance to a constant value that is not influenced by the amount of service accumulated for the domain. If you do not set a fixed contention index for a domain, the system determines the domain's contention index as follows:

- If the actual service rate for the domain is less than either the low ASRV or DSRV value, the contention index is greater than 100.
- If the actual service rate for the domain is in the specified ASRV or DSRV range, the contention index is between 1 and 100.

  The system calculates the contention index as follows:

  ```
   high service value - actual service
  ------------------------------------  * 100 = contention index
   high service value - low service value
  ```

  For example, if the ASRV was specified as (1000,3000) the high service value is 3000 and the low service value is 1000. Therefore, in our equation, if the domain is currently using 1500 service units, the contention index would be ((3000-1500) / 2000) * 100 = 75.
- If the actual service rate for the domain is greater than either the high ASRV or DSRV value, the contention index is less than 1.

If there are no out ready users, that domain will not be subject to an MPL increase.

The contention index is used to determine which domain is allowed to increase its MPL, and therefore allow additional address spaces into the multi-programming set. The contention index is also used to determine which domain is to decrease its MPL because of over-utilization.

## Dispatching Priorities

Once an address space has been swapped in, it competes with all other address spaces, regardless of domain, based on 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 and service rates.

You assign all dispatching priorities by using the DP parameter in the IPS. Because some address spaces provide services to other address spaces, you need to plan dispatching priorities carefully. The address space providing the service is called the server address space. The server should have a higher dispatching priority than the other address spaces. Then, if the system is constrained, the server will not be blocked out from processing the work for the other address spaces.

For example, suppose you are using a CICS monitor tool which you assign dispatching priority X'9A'. The tool provides some services to a CICS application region, which is assigne d dispatching priority X'9D' When the system is unconstrained, everything functions properly. When the CICS transaction load is heavy, most of the CPU is used in the CICS region, and the monitor tool never gets a chance to execute. The tool's work queue increases and consumes more and more CSA storage. To correct these problems, you should assign the monitor tool a dispatching priority of at least X'9E' so that it is higher than the CICS region.

## TSO/E Response Time Control

The TSO/E response time parameter (RTO) limits the service given to TSO/E and attempts to maintain consistent response time for similar TSO/E 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/E transaction response times. The function can reserve the excess capacity for future applications. The function cannot improve TSO/E response time in a heavily loaded system.

SRM enforces the response time objective by delaying a TSO/E 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/E 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/E 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/E 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, 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.

## IPS Example

Let us now collect many of the specifications used so far in this section into a complete IPS, using the syntax described in the *z/OS MVS Initialization and Tuning Reference*.

```
CPU=10.0,IOC=5.0,MSO=3.0,SRB=10.0

DMN=1,ASRV=(1000,2000)
DMN=2,ASRV=(1000,2000)
DMN=3,DSRV=(1,999999999)
DMN=4,DSRV=(1,999999999)

PGN=1,(DMN=4,DP=M2)

PGN=2,(DMN=1,DP=F30,DUR=400)
      (DMN=2,DP=F20,DUR=1600)
      (DMN=3,DP=M0)
```

The following observations can be made about transactions in this system:

- During Period 1, the ASRV parameter insures that the average service of address spaces is good and the transactions have a high fixed dispatching priority. Note again that each specification is in effect only for the duration of the respective period.

  With DUR=400, you are trying to give between .2 (400/2000) and .4 (400/1000) second response time during period 1.
- Transactions requiring more than 400 service units are then switched to domain 2 with a lower fixed dispatching priority.

  With DUR=1600, the goal is to complete within .8 to 1.6 additional seconds for a total response in the 1 to 2 second range.
- Transactions requiring more than 2000 service units (DUR=400 + DUR=1600) are then switched to domain 3 and a low mean-time-to-wait dispatching priority.
- 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.

The purpose of the examples in this section is to show concepts, not to serve as recommendations.

### SET IPS Command

The operator can use the SET command between IPLs to use a different IEAIPSxx member of SYS1.PARMLIB. 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 (2 for TSO/E users, 1 for other work).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 a TSO/E session or an executing non-privileged job by issuing the RESET jobname, PERFORM=nnnn command. (A privileged job runs in performance group zero.) 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 *z/OS MVS 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, which will be 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" on page 3-15 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/E user attribute dataset (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 transactionsaccording 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. 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.
- Report performance groups - to report statistics on thetransactions 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. For subsystems using workload management services, such as CICS and IMS, you can associate a report performance group with a service class defined in a workload management service policy. You can get response time information about the CICS and IMS transactions as they would be defined if the system were running in workload management goal mode.

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. The report performance group has no IPS entry.

### 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 account number, 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 nor the master scheduler address space. These are always controlled in performance group zero, a system-defined performance group.

The following subsystem types are subject to report PGN **and** control PGN assignment, because their transactions are under direct SRM control.

- STC subsystem (system-defined). The transactions for this subsystem include all work initiated by the START and MOUNT commands. STC also includes many system component address spaces, such as the TRACE, PC/AUTH, ALLOCAS, LLA, and VLF address spaces (for more information, see “Special Considerations for the STC Subsystem” on page 3-29).
- TSO/E subsystem (system-defined). The transactions for this subsystem include all commands issued from foreground TSO/E sessions.
- Job entry subsystem (usually JES2 or JES3). The transactions for this subsystem include all jobs that it initiates.
- ASCH subsystem. The transactions for this subsystem include all APPC transaction programs scheduled by the IBM-supplied APPC/MVS transaction scheduler.
- OMVS subsystem. The transactions for this subsystem include all forked address spaces.

The following subsystem types are also subject to report PGN **and** control PGN assignment. You assign a performance group to the work of these subsystems by associating them with service classes using the SRVCLASS parameter.

- DB2 subsystem. The transactions include all work requests (actually, just the remote pieces of a split query) that DB2 has split into multiple queries and spread across multiple images in a sysplex.
- DDF subsystem. The transactions include all DB2 distributed data facility (DDF) requests.
- LSFM subsystem. The transactions include all work for LAN Server for MVS.
- SOM subsystem. The transactions include all SOM requests.
- IWEB subsystem. The transactions include all Internet Connection Server requests.
- Any subsystem using enclaves. An enclave is a transaction that can span multiple tasks and/or SRBs in one or more address spaces, and is managed and reported as a unit. Refer to the subsystem’s reference information to determine if it uses enclaves and, therefore, supports the use of report and control performance groups.

The following subsystems are subject **only** to report PGN assignment, control PGNs **cannot** be assigned to them.

- CICS subsystem. The transactions include those reported through the workload management services or the transaction reporting SYSEVENTs. You can assign a report performance group to CICS by associating it with a service class using the SRVCLASS parameter.
- IMS subsystem. The transactions include those reported through the workload management services. You can assign a report performance group to IMS by associating it with a service class using the SRVCLASS parameter.

- Any subsystem using the IWMRPT or IWMMNTY service or the SYSEVENT macro to pass transaction information to SRM. Refer to the subsystem's reference information to determine if it uses these services and, therefore, supports the use of report performance groups.

The following is a sample installation control specification and a portion of its corresponding IPS:

```
IEAICSxx                    IEAIPSxx

SUBSYS=STC, PGN=1           PGN=1,(DMN=1,DP=M1,...)
SUBSYS=TSO, PGN=2           PGN=2,(DMN=2,DP=F14,...)
SUBSYS=JES2, PGN=3          PGN=3,(DMN=3,DP=M0,...)
SUBSYS=ASCH, PGN=4          PGN=4,(DMN=4,DP=F10,...)
SUBSYS=OMVS, PGN=5          PGN=5,(DMN=5,DP=F12,...)
```

The sample IEAICSxx member has five subsystem sections; each specifies a performance group that applies to the subsystem as a whole. All started tasks, including JES2 and the kernel , are controlled and reported on in performance group 1; all TSO/E users in performance group 2; all JES2 jobs in performance group 3; all APPC transaction programs in performance group 4; and all forked address spaces in performance group 5.

## Transaction Entries

Within each subsystem section, the installation control specification entries can identify transactions in the following ways:

- By accounting information (ACCTINFO keyword)
- By transaction name (TRXNAME keyword)
- By userid (USERID keyword)
- By transaction class (TRXCLASS keyword)
- By service class (SRVCLASS keyword).

Not all keywords are significant for each subsystem. The following table defines the installation control specification keywords and indicates whether the keywords are significant, 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 | ACCTINFO -Accounting information | Yes | Yes |
| | TRXNAME - The TSO/E command name[1] | No | Yes |
| | USERID - The userid specified at LOGON | Yes | Yes |
| | TRXCLASS - N/A | N/A | N/A |
| STC | ACCTINFO | Yes | Yes |
| | TRXNAME - The name specified on the START command or the name of the system address space (see "Special Considerations for the STC Subsystem" on page 3-29). | Yes | Yes |
| | USERID | Yes | Yes |
| | TRXCLASS - N/A | N/A | N/A |

| Subsystem | Keyword | Control PGN (PGN) | Report PGN (RPGN) |
|---|---|---|---|
| JES2[2] | ACCTINFO - Accounting Information | Yes | Yes |
| | TRXNAME - The jobname of the JES2-initiated job | Yes | Yes |
| | USERID - The userid specified on the JOB statement through the RACF USER keyword | Yes | Yes |
| | TRXCLASS - The job class used for work selection | Yes | Yes |
| JES3[2] | ACCTINFO - Accounting Information | Yes | Yes |
| | TRXNAME - The jobname of the JES3-initiated job | Yes | Yes |
| | USERID - The userid specified on the JOB statement through the RACF USER keyword | Yes | Yes |
| | TRXCLASS - The job class used for work selection | Yes | Yes |
| ASCH | ACCTINFO - Accounting Information | Yes | Yes |
| | TRXNAME - The jobname of the JCL JOB statement in the APPC/MVS transaction program (TP) profile. | Yes | Yes |
| | USERID - The userid of the user requesting the APPC/MVS service. | Yes | Yes |
| | TRXCLASS - The class in the APPC/MVS transaction program (TP) profile. | Yes | Yes |
| OMVS | ACCTINFO - Accounting Information. If TAILOR_ACCOUNT(YES) is specified in the TP profile, the account number is inherited from the parent. Otherwise, the account number in the TP profile JOB statement is used. | Yes | Yes |
| | TRXNAME - The jobname for the OMVS address space. | Yes | Yes |
| | USERID - The userid of the forked child, inherited from the parent. | Yes | Yes |
| | TRXCLASS - Not applicable. | N/A | N/A |
| CICS[3] | SRVCLASS | No | Yes |
| DB2[4] | SRVCLASS | Yes | Yes |
| DDF[5] | SRVCLASS | Yes | Yes |
| IMS[6] | SRVCLASS | No | Yes |
| IWEB[7] | SRVCLASS | Yes | Yes |
| LSFM | SRVCLASS | Yes | Yes |
| SOM | SRVCLASS | Yes | Yes. |
| ANY[8] | ACCTINFO - Accounting Information | No | Yes |
| | TRXNAME - The transaction name (usually command name) | No | Yes |
| | USERID - The userid specified at LOGON. | No | Yes |
| | TRXCLASS - The transaction class | No | Yes |
| | SRVCLASS - The service class | No | Yes |

| Subsystem | Keyword | Control PGN (PGN) | Report PGN (RPGN) |
|---|---|---|---|
| **Footnotes** | | | |
| [1] | The TSO/E LOGON command is not reported on through the installation control specification unless LOGON is used to terminate a TSO/E session. TSO/E subcommands are reported under the primary command name. Commands within a TSO/E CLIST are reported on under the EXEC command. | | |
| [2] | The job entry subsystem name is the primary or alternate subsystem name specified in the IEFSSNxx parmlib member. This table and the remainder of this section assume the name is either JES2 or JES3. | | |
| [3] | Must be at least CICS/ESA Version 4.1 | | |
| [4] | Must be at least DB2 Version 5 | | |
| [5] | Must be at least DB2 Version 4.1 | | |
| [6] | Must be at least IMS/ESA Version 5 | | |
| [7] | Must be at least Internet Connection Server (ICS) Version 2.2 | | |
| [8] | For illustration, this section assumes that a hypothetical user-written subsystem, ANY, invokes either the transaction reporting SYSEVENTs or the workload management services. This interactive subsystem is included only as an example. Refer to the current documentation for an IBM or user-written subsystem to determine if the subsystem used either the SYSEVENTs or the workload management services and, if it does, what the subsystem name is and which keywords are valid. | | |

The following sample ICS and a portion of its corresponding IPS show the use of transaction entries within a subsystem:

```
IEAICSxx                          IEAIPSxx

MASK=*
SUBSYS=JES2,PGN=3                 PGN=3(DMN=3,...)
   ACCTINFO=D58***(2),PGN=30      PGN=30(DMN=3,...)
   TRXNAME=PAYROLL,PGN=31         PGN=31,(DMN=3,...)
   USERID=D58RPM1,PGN=32          PGN=32,(DMN=3,...)
   TRXCLASS=AB,RPGN=33
```

All jobs with the account numbers starting with D58 (in the second position) followed by three characters will be assigned to performance group 30. PAYROLL is controlled in performance group 31. All other jobs that specify userid D58RPM1 in their JCL are controlled in performance group 32. All remaining JES2 jobs are controlled in performance group 3. Also, all statistics for TRXCLASS=AB jobs (even PAYROLL if it is TRXCLASS=AB) are reported on in 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 significant for report performance groups because multiple report PGNs 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 ACCTINFO.

     ACCTINFO
          If the account number matches an ACCTINFO entry and
          the entry specified a control PGN, this PGN is assigned
          to the transaction.  Otherwise, 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.

               USERID
                    If the transaction user id matches a USERID
                    entry and the entry specifies 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 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, ASCH, OMVS, and job entry
                                 subsystem = performance group 1

                                 TSO/E subsystem = 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.
```

## Special Considerations for the STC Subsystem

STC work includes many system component address spaces and work initiated by the START and MOUNT commands. Depending on certain factors, SRM provides special handling for the various types of work that run under the STC subsystem. The sections that follow describe how SRM handles different types of STC work.

***Work Specified as Privileged in the PPT or SCHEDxx:*** Unless explicitly assigned to a performance group in the ICS, SRM automatically assigns to performance group zero all work that is specified as privileged through the program properties table (PPT) or in the SCHEDxx parmlib member. SRM ensures this work remains swapped in, except during long waits. To assign a dispatching priority to this work, SRM uses the value specified on the PVLDP parameter in IEAIPSxx.

For example, NetView Performance Monitor (NPM) is privileged in the IBM-supplied PPT. If there is no performance group assignment found for NPM in the ICS, SRM assigns NPM to performance group zero by default.

Only SRM can assign work to performance group zero.You can assign privileged work to a non-zero performance group. You must explicitly assign the privileged work to a PGN other than the STC subsystem default in the ICS. That is, you must include in the ICS under SUBSYS=STC the TRXNAME of the started task, and its assigned PGN. Once you control it through a non-zero performance group, you own it, and can treat it as you do any other work. For example, you can use the RESET command on the started task. The privileged attribute is lost for the duration of the started task, so assign it an appropriate performance group.

If you are controlling the work (in a non-zero performance group), you cannot use the RESET command to move the work to performance group zero. If you decide you no loner want control, and would like the system to control the work in performance group zero, you must remove the performance group assignment from the ICS and re-start the started task.

**Note:** Some started tasks require an IPL to re-start.

You can determine which system component address spaces are privileged through RMF Monitor II (in the ASD report). Privileged system component address spaces are those listed as having PGN 0, Domain 0, and the X'FF' dispatching priority (DP). Examples of privileged system component address spaces are:

***MASTER***
: Master address space

**CATALOG**
: Catalog services

**SMF** System management facilities

***Other (Non-Privileged) System Component Address Spaces:*** If you do not assign some system component address spaces to a control performance group, then they are automatically assigned to the X'FF' dispatching priority. Having the highest dispatching priority as the default for these address spaces saves installations from having to add entries to the installation control specification whenever new system component address spaces are added (for example, with later releases of MVS or related program products).

You can determine which system component address spaces are automatically assigned to the X'FF' dispatching priority through RMF Monitor II (for example, in the ASD report). In the RMF report, find the address spaces that are assigned **all** of the following attributes:

- Non-zero performance group
- Non-zero domain
- X'FF' dispatching priority

Those address spaces are system component address spaces. Examples of these address spaces include:

**ALLOCAS**
: Allocation Services and data areas

**ANTMAIN**
: DF/SMS concurrent copy address space

**CONSOLE**
: Communications task

**RASP** Real storage manager (RSM)

**XCFAS**
Cross-system coupling facility (XCF)
**IOSAS**
Input/output supervisor (IOS)
**SMXC** DF/SMS address space supporting PDSEs.
**WLM** Workload manager address space

By default, the control PGN for these system address spaces is the PGN that was specified for the STC subsystem in the IEAICSxx parmlib member (for example, SUBSYS=STC,PGN=5). These address spaces can be assigned control PGNs by name under the STC subsystem name using the TRXNAME keyword in IEAICSxx. They then lose the X'FF' dispatching priority and pick up the value specified by the assigned PGN.

***Remaining STC Work:*** Work that does not match any of the preceding descriptions can be assigned a control performance group through the installation control specification. As with the category of work described under "Other (Non-Privileged) System Component Address Spaces" on page 3-30, the default control performance group for these system address spaces is the PGN that was specified for the STC subsystem in the IEAICSxx parmlib member. These address spaces can be assigned control PGNs by name under the STC subsystem name using the TRXNAME keyword in IEAICSxx.

This group includes the following system component address spaces:
**PCAUTH**
Cross memory authorization
**TRACE**
System trace
**LLA** Library lookaside
**VLF** Virtual lookaside facility
**SMS** Storage management subsystem
**SYSBMAS**
System buffer management subsystem
**OMVS** Kernel address space

***Setting Dispatching Priorities for LLA and VLF:*** Setting of a high dispatching priority for the LLA and VLF address spaces will reduce the time required to initialize or refresh LLA and VLF. LLA and VLF should not, however, run at a higher dispatching priority than the IMS control region or a CICS terminal owning region. 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.

## Special Considerations for Subsystems Using Enclaves

You can use control and report performance groups for any subsystem using enclaves. IBM-supplied subsystems using enclaves are DDF, DB2, IWEB, LSFM, and SOM. In this section, the term ***subsystem*** is used to refer to a subsystem using enclaves.

Assigning a control or report PGN to a subsystem's enclave work requires using the SRVCLASS keyword in the IEAICSxx member and also requires an active workload management service policy. When a subsystem transaction begins, the classification rules in the active service policy are used to assign it to a service class. The SRVCLASS keyword maps this service class to a PGN.

The steps needed to assign a PGN to the subsystem work are:

- Using the WLM application, define a service policy, with one or more service classes representing the work. Define classification rules for the service classes under the applicable subsystem type.
- In your IEAIPSxx parmlib member, decide which performance group you would like to control the subsystem work. You can use an existing performance group, or define a new one.
- In your IEAICSxx parmlib member, under the subsystem type, set the SRVCLASS parameter to the service class defined in the service policy, and associate it with the performance group.

  **Note:** Make sure you assign a fixed dispatching priority for an enclave performance group that is less than or equal to the dispatching priority of the address space(s) where the enclave work executes. If the enclave dispatching priority is greater than that of the address space(s) where the enclave work executes, you will get unpredictable results.
- Put the updated IEAIPSxx and IEAICSxx members into effect with the SET command.
- Install the service definition and activate the service policy.

All work in that service class is associated with the PGN, and processed according to the PGN's controls. If the above steps were not completed and SRM cannot assign a performance group, then the subsystem work is assigned the performance group of the subsystem's address space.

**Example of Assigning a Control PGN to DDF**

The following example shows how you can use the SRVCLASS keyword to assign a control performance group to DDF transactions:

- In your service definition, set up the following:
  - A test policy:

    ```
    Policy:       TEST
    Description:  Test policy
    ```
  - A service class for DDF transactions:

    ```
    Service Class: DDFALL
    Description:   All DDF transactions
    Goal:          5 seconds average
    ```

    **Note:** The response time goal assigned is irrelevant, since the system is running in compatibility mode.
  - Classification rules for the DDF subsystem:

    ```
    Subsystem Type:        DDF
    Default Service Class: DDFALL
    ```
- In your IEAIPSxx member:

  ```
  PGN=106,(DP=F14,...)
  ```
- In your IEAICSxx member:

  ```
  SUBSYS=DDF,
   SRVCLASS=DDFALL,PGN=106
  ```
- Issue a SET ICS command to set the changed IEAICSxx member.
- Activate the *test* service policy by issuing the following command:

  ```
  VARY WLM,POLICY=TEST
  ```

Your DDF work is then processed according to the controls defined for PGN 106.

## Optional Control Performance Groups

An installation can include batch jobs with multiple steps or TSO/E users 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” on page 3-80.)

Any IEAICSxx entry that specifies a control PGN can also specify one or more optional control performance groups (OPGNs). 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. 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 the workload management services, the SYSEVENT macro and for TSO/E 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 entry types (SUBSYS, ACCTINFO, TRXNAME, USERID, SRVCLASS, and TRXCLASS).

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/E, STC, ASCH, JES2, or JES3 subsystems, report performance groups contain response time and service usage data. For all other subsystems, refer to their documentation to determine whether service usage data is passed to the SYSEVENT macro.

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 five report performance groups, one for each type of entry. For the SRVCLASS parameter however, you can not combine the report performance group with any other 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/E userid and a TSO/E command. However, the same RPGN can be specified for multiple entries of the same type. Also, the same RPGN can be

specified in more than one subsystem section so that the report spans several subsystems. For example, the VTAM started task can be included in a total TSO/E 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/E, ASCH, or JES). 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.
- Also, 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.

## Using Report Performance Groups to Prepare for Goal Mode

While your system is running in workload management compatibility mode, you can prepare for using goal mode with the SRVCLASS keyword in the ICS. This parameter lets you associate a service class with a report performance group to get response time information to help you set up service goals for transactions. With the SRVCLASS keyword, you can get response time information for transactions as they would be processed in goal mode. The SRVCLASS parameter applies to subsystem types supporting the workload management services. For SP 5.1, these are CICS/ESA Version 4.1 and IMS/ESA Version 5.

You define a service policy using the WLM ISPF application. In that service policy, define service classes representing the transactions for which you want the response time information. Define classification rules for the service classes. In your ICS, under the subsystem type, set the SRVCLASS parameter to the service class defined in the policy. You then install the service definition and activate the service policy. All work in that service class is reported under the report performance group in the RMF Monitor I Workload Activity Report. The information reported is the number of completed transactions, and their average response time. The following example shows how you can use the SRVCLASS keyword.

For CICS response time information while in compatibility mode, you can set up the following:

- In your service definition, set up the following:
  - A test policy:

    ```
    Policy:      TEST
    Description: Policy for reporting information
    ```

  - A service class for CICS transactions:

```
Service Class: CICSALL
Description:   All CICS transactions
Goal:          5 seconds average
```

**Note:** The response time you assign is irrelevant, since the system is running in compatibility mode.

– Classification rules for the CICS subsystem:

```
Subsystem Type:        CICS
Default Service Class: CICSALL
```

- In your IEAICSxx member:

```
SUBSYS=CICS,
 SRVCLASS=CICSALL,RPGN=100
```

- Issue a SET ICS command to set the changed IEAICSxx member.
- Activate the *test* service policy by issuing the following command:

```
VARY WLM,POLICY=TEST
```

You then receive response time information about CICS transactions in the RMF Monitor I Workload Activity Report under report performance group 100.

For more information about using this keyword, see *z/OS MVS Planning: Workload Management*.

## 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/E by department or some other user grouping, assuming conventions are followed for generating user ids.

*Example 1:* Assume that all TSO/E 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/E users are controlled in performance group 2. In addition, statistics for department D09 users are collected in reporting 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/E commands and their aliases.

```
SUBSYS=TSO,PGN=2
 TRXNAME=EX(1),RPGN=21
```

In this example, statistics for all TSO/E 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 ACCTINFO, one TRXNAME, one USERID, and one TRXCLASS 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 characters; 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=9GES(3),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 D09BRP11 is controlled in PGN 5
- Userid D09JMB1 is controlled in PGN 7
- Userid D08SFS1 is controlled in PGN 3

**Note:** Remember that all userids are eight characters long (this includes possible trailing blanks). Therefore, USERID=D09(1) specifies three characters in positions one, two and three. On the other hand, USERID=D09%%%% specifies four characters because the eighth character is not masked out and must therefore be a blank for you to get a match for the PGN assignment.

*Example 2:* To report a TSO/E 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/E 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.

## SET ICS Command

The operator can use the SET command between IPLs to change to another IEAICSxx parmlib member. If the system was initialized 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. If however, an MVS RESET command that specifies the PERFORM keyword has previously been issued to change the performance group number of a transaction, SET ICS processing does not change that transaction's PGN again. It retains the value the RESET command gave it. 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 *z/OS MVS System Commands*.

## Installation Control Specification Examples

This topic contains two examples of an installation control specification and the related portions of its corresponding IPS.

*Example 1:*

| **IEAICSxx** | **IEAIPSxx** |
|---|---|
| SUBSYS=TSO,PGN=2 | PGN=2,(DMN=,DP=,...) |
| SUBSYS=STC,PGN=5 | PGN=5,(DMN=,DP=,...) |
| TRXNAME=IEEVMPCR,PGN=6 | PGN=6,(DMN=,DP=,...) |
| SUBSYS=JES2,PGN=1 | PGN=1,(DMN=,DP=,...) |

This example shows the assignment of control performance groups based on the type of work (TSO/E, started task, mount, batch). Specifying a PGN for TSO/E allows for the automatic override of a PERFORM parameter on the LOGON command. Specifying the TSO/E subsystem in the installation control specification eliminates the need to check for optional PGNs in the UADS dataset. All TSO/E 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<br>TRXNAME=GETNEXT,RPGN=13 | |

```
   TRXNAME=INQUIRE,RPGN=14
   TRXNAME=REPORT,RPGN=15
   TRXCLASS=255,RPGN=16
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=DUMPSRV,PGN=17                          PGN=17,(DMN=,DP=,...)
   TRXNAME=IEEVMPCR,PGN=1
SUBSYS=JES3
   ACCTINFO=D58(2),PGN=90                          PGN=90,DMN=,DP=,...)
   TRXNAME=PAYROLL,PGN=91                          PGN=91,(DMN=,DP=,...)
   TRXNAME=MRBIG,PGN=91
   TRXCLASS=A,PGN=92                               PGN=92,(DMN=,DP=,...)
   TRXCLASS=S,PGN=93                               PGN=93,(DMN=,DP=,...)
   TRXCLASS=T,PGN=94                               PGN=94,(DMN=,DP=,...)
SUBSYS=ASCH,PGN=4                                  PGN=4,(DMN=,DP=,...)
   ACCTINFO=D001P024,PGN=41                        PGN=41,(DMN=,DP=,...)
   TRXNAME=MAIL,PGN=44
   TRXNAME=LOG,PGN=44                              PGN=44,(DMN=,DP=,...)
   USERID=SPECIAL2,PGN=47                          PGN=47,(DMN=,DP=,...)
   TRXCLASS=Y,PGN=49                               PGN=49,(DMN=,DP=,...)
```

This example shows the assignment of control and report PGNs based on ACCTINFO, TRXNAME, USERID, TRXCLASS, 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 the SYSEVENT macro. Reporting is assigned based on the transactions' names and classes. Response times for transactions with names of GETNEXT, INQUIRE and REPORT are accumulated in reporting performance groups 13, 14, and 15, respectively. All transactions, including the three preceding names, 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 reporting 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 SYSEVENT macro instruction 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/E users are controlled by performance group 40. The assignment of TSO/E users to installation specified performance groups ensures that the execution control parameters assigned to the users meet the installation's requirements.

**Notes:**

1. 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/E 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. The installation can report on TSO/E 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 PGN 10 to the IMS control region. The message processing regions (MPRs) have unique jobnames MPR1, MPR2, and so forth, and they are assigned to PGN 11. The special system address space DUMPSRV is assigned to PGN 17. Once you assign a PGN to a special system address space under SUBSYSTEM=STC with the TRXNAME= keyword, the address space loses its special system attributes and is controlled by the PGN attributes. 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, those account numbers with D58 in the second position are assigned to performance group 90. The PAYROLL and MRBIG jobs are assigned to performance group 91. Jobs belonging to JES3 classes A, S, or T are assigned performance groups 92, 93, or 94, respectively. All other batch jobs are assigned to performance group 1, the default specification for non-TSO/E 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.

**SUBSYSTEM ASCH**

If initiated by the IBM-supplied APPC/MVS transaction scheduler (ASCH), those transactions with account number D001P024 are assigned to performance group 41. All MAIL and LOG transactions are assigned to performance group 44.

An APPC transaction program submitted by a person whose userid is SPECIAL2 runs in performance group 47. Any transaction in CLASS Y that did not fall into any other category is in performance group 49.

# OPT Concepts

The parameters discussed in this part are explained in *z/OS MVS Initialization and Tuning Reference*.

See “Section 3: Advanced SRM Parameter Concepts” on page 3-41 for information about advanced OPT concepts.

## MPL Adjustment Control

The OPT provides keywords to specify upper and lower thresholds for the variables that SRM uses to determine if it should increase, decrease, or leave the MPL unchanged. 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 can influence a change.

| Control variable and internal name | Thresholds that can influence an MPL change: | | Keyword in OPT |
|---|---|---|---|
| | decrease | increase | |
| CPU utilization (RCVCPUA) | >RCCCPUTH | <RCCCPUTL | RCCCPUT |
| Page fault rate (RCVPTR) | >RCCPTRTH | <RCCPTRTL | RCCPTRT[1] |
| UIC (RCVUICA) | <RCCUICTL | >RCCUICTH | RCCUICT |
| Percentage of online storage fixed (RCVFXIOP) | >RCCFXTTH | <RCCFXTTL | RCCFXTT |
| Percentage of storage that is fixed within the first 16 megabytes (RCVMFXA) | >RCCFXETH | <RCCFXETL | RCCFXET |

[1]The default thresholds for this keyword causes the corresponding control variable to have no effect on MPL adjustment.

## Other Options

The following additional options can also be specified in the OPT.

***Transaction Definition for CLISTs:*** An installation can specify whether the individual commands in a TSO/E CLIST are treated as separate TSO/E commands for transaction control. 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.

***Directed VIO Activity:*** VIO dataset pages can be directed to a subset of the local paging datasets through *directed VIO*, which allows the installation to direct VIO activity away from selected local paging datasets 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, with the PAGE system parameter, allows the installation to define those local paging datasets that are not to be used for VIO, leaving the rest available for VIO activity. However, if space is depleted on the paging datasets made available for VIO paging, the non-VIO paging datasets 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.

***Alternate Wait Management:*** An installation can specify whether to activate or deactivate alternate wait management (AWM). If AWM is activated, SRM and LPAR cooperate to reduce low utilization effects and overhead.

Specifying CCCAWMT with any value in the range 1 to 499999 makes AWM active. Specifying CCCAWMT with any value in the range 500000 to 1000000 makes AWM inactive. The default is 12000 (AWM active).

## Section 3: Advanced SRM Parameter Concepts

This section discusses some advanced 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 *z/OS MVS Initialization and Tuning Reference* and should be referred to when necessary.

IBM recommends not changing these values unless you have a particular installation requirement that demands a change.

## I/O Priorities

If the I/O priority queuing function is requested in the IPS, I/O requests for users 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:

   DP=M0,IOP=M1

   the I/O priority would be fixed. In this case, because IOP=M1 was specified, the priority range would be from 10-19. Because this is an I/O priority, however, the second digit of the priority is always zero. Therefore, the fixed priority in this

example is 10. (If IOP=M0 had been specified, the fixed priority would have been 00; if IOP=M2 had been specified, the fixed priority would have been 20.)

## 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 migrates or steals frames that reduce the application's working set below the critical number, the application's performance is affected. 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 affected.

In systems with expanded storage, the working set of an application or the common area includes all the frames allocated in processor storage. Processor storage includes central and expanded storage. The system disregards the target working set size when replacing pages in central storage and moving those pages to expanded storage. When a page is migrated 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. Storage isolation applies to an address space and the data spaces, hiperspaces, and VIO pages owned by 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 stealing 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 if the address space had 60 or more frames allocated when it was swapped out. If you specify a maximum working set size that is less than the job needs to run, you may experience unwanted paging.

### Possible Uses of Storage Isolation

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/E response time for trivial transactions is important. With sufficient central 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 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 two page groups for local page data sets on a 3380; a 55-page working set also requires an allocation of two page groups for local page data sets on a 3380.

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 has 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.

## Considerations When Using Storage Isolation

In deciding if storage isolation should be used to protect an application, consider the following:

- 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 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 (central and expanded storage).
- SRM will not manage an address space that is storage isolated. See “Working Set Management” on page 3-9 for information on how SRM manages address spaces.
- 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 central 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/E performance objectives.

If the installation determines that storage isolation is necessary for an application, the application's use of central storage should be examined. Using RMF or SMF data and other application data, 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.

## 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.

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.

A processor that enters a wait state is always enabled for I/O interruptions, however, regardless of what you specify for the CPENABLE keyword.

In addition to enabling a processor when I/O activity requires it, SRM also enables another processor for I/O interruptions if one of the following occurs:
- An enabled processor is taken offline
- An enabled processor has a hardware failure
- SRM detects that no I/O interruptions have been taken for a predetermined period of time and concludes that the enabled processor is unable to accept interrupts.

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.

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 | | Keyword in OPT |
|---|---|---|---|
| | under | over | |
| Percentage of I/O interruptions through TPI instruction (ICVTPIP) | <ICCTPILO | >ICCTPIHI | CPENABLE |

## Time Slice Functions

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.

**Note:** IBM no longer recommends SRM-managed time slicing. Changes in the dispatcher have eliminated the need for it in most cases.

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.

The time slice function allows a job to have two differentpriorities 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.

Table 3-1 relates SRM seconds to real time. The SRM constants that are shown in this table are merely generalizations and approximations. For more accurate comparisons of processors, see the internal throughput rate (ITR) numbers in *Large Systems Performance Reference (LSPR)*, SC28-1187.

*Table 3-1. Relating SRM Seconds to Real Time*

| Processor Model | SRM Seconds/Real Seconds |
|---|---|
| **Processors: zSeries 900** | |

*Table 3-1. Relating SRM Seconds to Real Time (continued)*

| Processor Model | SRM Seconds/Real Seconds |
|---|---|
| zSeries 900 Models 101–109 | 269.3964 |
| zSeries 900 Models 110–116, 1C1–1C9 | 281.5314 |
| **Processors: S/390 9672** | |
| S/390 9672 R2 (All Models) | 23.9280 (per CP) |
| S/390 9672 R3 (All Models) | 25.3447 (per CP) |
| S/390 9672 R4 (All CPs) | 52.6593 |
| S/390 9672 Models R15 to R45 | 68.3060 |
| S/390 9672 Models R55 to RX5 (All Models) | 72.7590 |
| S/390 9672 Models RA5 and RB5 | 53.7634 |
| S/390 9672-RC5 (All Models) | 60.9756 (per CPU) |
| S/390 9672-RY5 (All Models) | 81.3008 |
| **Processors: S/390 9672 G5 Models** | |
| S/390 9672-R16 | 129.9376 |
| S/390 9672-R26 | 129.9376 |
| S/390 9672-R36 | 141.0834 |
| S/390 9672-R46 | 141.0834 |
| S/390 9672-R56 | 141.0834 |
| S/390 9672-R66 | 141.0834 |
| S/390 9672-R76 | 141.0834 |
| S/390 9672-R86 | 141.0834 |
| S/390 9672-R96 | 141.0834 |
| S/390 9672-RA6 | 97.9623 |
| S/390 9672-RB6 | 97.9623 |
| S/390 9672-RC6 | 129.9376 |
| S/390 9672-RD6 | 129.9376 |
| S/390 9672-RX6 | 141.0834 |
| S/390 9672-T16 | 141.0834 |
| S/390 9672-T26 | 141.0834 |
| S/390 9672-Y16 | 168.4635 |
| S/390 9672-Y26 | 168.4635 |
| S/390 9672-Y36 | 168.4635 |
| S/390 9672-Y46 | 168.4635 |
| S/390 9672-Y56 | 168.4635 |
| S/390 9672-Y66 | 168.4635 |
| S/390 9672-Y76 | 168.4635 |
| S/390 9672-Y86 | 168.4635 |
| S/390 9672-Y96 | 168.4635 |
| S/390 9672-YX6 | 168.4635 |
| **Processors: S/390 2003** | |
| S/390 2003 Model 107 | 27.8520 |
| S/390 2003 Model 124 (All Models) | 30.3988 (per CPU) |
| S/390 2003 Model 1C5 (All Models) | 37.6279 (per CPU) |
| S/390 2003 Model 2X7 (All Models) | 46.0914 (per CPU) |
| S/390 2003 Model 203 | 6.1814 |
| S/390 2003 Model 204 | 10.3203 |
| S/390 2003 Model 205 | 14.4375 |
| S/390 2003 Model 206 | 18.5680 |
| S/390 2003 Model 207 | 27.8520 |
| S/390 2003 Model 215 | 33.3333 |
| S/390 2003 Model 216 | 41.6666 |
| S/390 2003 Model 224 (All Models) | 30.5325 (per CPU) |

*Table 3-1. Relating SRM Seconds to Real Time (continued)*

| Processor Model | SRM Seconds/Real Seconds |
|---|---|
| S/390 2003 Model 225 (All Models) | 39.9872 (per CPU) |
| S/390 2003 Model 246 (All Models) | 41.1455 (per CPU) |
| S/390 2003 Model 2C5 (All Models) | 37.6279 (per CPU) |

# 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. The following functions can be modified by parameters in the OPT:

- Enqueue residence control
- SRM invocation interval control
- Logical swapping controls
- CPU Management control
- Pageable storage control
- Central storage control
- Expanded storage control
- Swap rate control

## 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” on page 3-10.

## 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.

## Logical swapping control

Four keywords are provided in the IEAOPTxx parmlib member to influence logical swapping:

1. LSCTMTE specifies the upper and lower values for the system think time limit.
2. LSCTUCT specifies the upper and lower UIC threshold values.
3. LSCTFTT specifies the upper and lower percentages of online central storage that is fixed.
4. LSCTFET specifies the upper and lower percentages of central storage that is fixed within the first 16 megabytes.

SRM uses keywords 2, 3, and 4 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 | %Think time limit | %Upper value | Keyword |
|---|---|---|---|
| LSCTMTEL | LSCTMTES | LSCTMTEH | LSCTMTE |

| Lower value | %Think time limit | %Upper value | Keyword |
|---|---|---|---|

- For think time change:

| Control variable and internal name | Think time change: decrease | Think time change: increase | Keyword in OPT |
|---|---|---|---|
| UIC (RCVUICA) | <LSCTUCTL | >LSCTUCTH | LSCTUCT |
| Percentage of online storage fixed (RCVFXIOP) | >LSCTFTTH | <LSCTFTTL | LSCTFTT |
| Percentage of storage that is fixed within the first 16 megabytes (RCVMFXA) | >LSCTFETH | <LSCTFETL | LSCTFET |

## CPU management control

The keyword provided for CPU management control influences the assignment of dispatching priorities in the mean-time-to-wait range.

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 relating to mean time to wait processing. 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 mean-time-to-wait dispatching.

Look at the RMF ASD report to decide whether you need to adjust the CCCSIGUR value. If all mean-time-to-wait are skewed toward the bottom, you need a larger CCCSIGUR value. If all mean-time-to-wait are skewed toward the top, you need a smaller CCCSIGUR 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. 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 RCETOTFX[1] | >MCCFXTPR | MCCFXTPR |
| Percentage of storage that is fixed within the first 16 megabytes (RCEBELFX)[1] | >MCCFXEPR | MCCFXEPR |

| Control variable and internal name | Shortage of pageable storage exists | Keyword in OPT |
|---|---|---|

[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.

## Central Storage Control

This parameter, specified by the MCCAFCTH keyword, indicates the number of frames on the available frame queue when stealing begins and ends. The range of values on this keyword determines the block size that SRM uses for stealing. In order to get a block into central storage, the lower value of the range must be greater than the block size.

## Expanded storage control

The criteria age will influence if a page that is being page out of central storage should be sent to expanded storage or to auxiliary storage. To modify the criteria age value, 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 age table entries.

**Note:** Expanded storage is only supported in ESA/390 mode. However, the ESCTVIO and ESCTBDS keywords still have meaning and affect storage management in z/Architecture mode.

This is a list of the OPT keywords, the allowed values of the index, and the reasons for page movement for which a criteria age can be specified:

| Keyword | Index Values | Page type |
|---|---|---|
| ESCTBDS | 3-99 or none | Hiperspace (block-addressable) pages |
| ESCTPOC | 0-99 | Page-out pages |
| ESCTSTC | 0-99 | Stolen pages |
| ESCTSWTC | 0-99 | Swap trim pages |
| ESCTSWWS | 0-99 | Swap working set pages ready for swap-out |
| ESCTVIO | 3-99 or none | VIO swap-out pages |

The ESCTPOC, ESCTSTC, ESCTSWTC, and ESCTSWWS keywords require that an index be specified to associate the types of paging groups. The possible values for the indices are:

**0** Privileged or nonswappable address spaces, or from the common area

**1** All others not in type 0 or 2

**2** Long, detected, and terminal wait swap, TSO/E stolen or paged-out, or APPC address space waits

**3-99** A user-specified value that is associated with a domain by using the ESCRTABX parameter of the IEAIPSxx member of SYS1.PARMLIB. If you do not specify the ESCRTABX parameter in your IPS, the system assigns a value based on the type of frame.

SRM 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 central storage. The system-high UIC measures how long (in seconds) a page has remained unreferenced in central storage. The system-high

UIC is actually an inverse measure of central storage contention. That is, when contention for central 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 central 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 contentionof 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.

SRM directs RSM to swap out trim pages to expanded storage when the sum of the system-high UIC plus the migration age is greater than the criteria age. RSM sends 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 plus the user think time. RSM sends changed and unchanged stolen pages, virtual I/O pages, hiperspace pages, and page-out requested pages to expanded storage when the migration age is greater than the criteria age.

Though these algorithms are used to determine when a page should go to expanded or auxiliary storage, if there is plenty of expanded storage available, SRM directs RSM to send pages directly to expanded storage without using these algorithms. If you would like to bypass the use of expanded storage, you can specify a criteria age of 32767.

There are a few special cases, however, when SRM doesn't use this criteria age specification to determine when to send pages to expanded or auxiliary storage. When an address space is storage isolated using the PWSS, PPGRT, or PPGRTR parameters, stolen pages, VIO pages, and hiperspace pages are sent to either expanded or auxiliary storage according to the following criteria:

- For stolen pages -

  Pages are sent to expanded storage unless none is available, in which case pages are sent to auxiliary storage.
- For VIO and hiperspace pages -
  - Pages are sent to expanded storage when the number of pages contained in processor storage (central plus expanded) is less than the PWSS minimum value.
  - Pages are sent to auxiliary storage when both of the following are true:
    - The number of pages contained in processor storage is greater than the PWSS maximum value
    - Expanded storage is constrained (nearly full).

  SRM uses the criteria age specification for storage isolated address spaces when any of the following are true:
  - The number of pages in processor storage is between the PWSS minimum and maximum values
  - The PPGRT or PPGRTR parameter is specified without a PWSS value
  - The number of pages contained in processor storage is greater than the PWSS maximum value and expanded storage is not constrained.

The criteria age values 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.

***Swap Rate Control:*** This parameter, specified by the SWAPRSF keyword, indicates how heavily to weigh the cost of doing an exchange swap in the system. The smaller the value, the more readily SRM will perform a swap. If too much exchange swapping is being done in the system, you can increase this value.

# Section 4: 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, 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 more than 30,000 units.

*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/E

*What is the number of terminals?*

*What is the average number of ready users?*

As a guideline for installations new to TSO/E, 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/E session (data entry, problem solving, program development).

*What is the required response time and expected transaction rate for different categories of TSO/E 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/E 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/E 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/E commands use 200 service units or less.
Medium length commands use between 200 and 1000 service units.
Long TSO/E commands use 1000 service units or more.

*What is the required service rate for TSO/E users?*

If 2-second response time (as reported by RMF) is required for very short TSO/E 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/E, 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?*

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” on page 3-71 in this section.
- 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 #4 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.

**Worksheet Number 1 - Service Definition Coefficients and Domain Information**

**CPU** _____

**IOC** _____

**MSO** _____

**SRB** _____

| Domain Definitions | | | | |
|---|---|---|---|---|
| **Domain Number** | **Minimum MPL** | **Maximum MPL** | **Service for Domain (ASRV or DSRV)** | **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 | | |

MTW - Mean-time-to-wait

[1] If different from dispatching priority.

**Worksheet Number 3 - Execution Characteristics Definition**

| Group | Period | Duration | Comments |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

**Worksheet Number 4 - 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= | | | |
| | USERID = | | | |
| | ACCTINFO= | | | |
| 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 5 - Performance Group/Period Definition**

| Performance Group Number | | Domain Number | Service | Dispatching Priority | | | I/O Priority | Duration | UNT |
|---|---|---|---|---|---|---|---|---|---|
| | | | | DP | TSDP | TSGRP | | | |
| Defaults → | | See Note 1 | ASRV DSRV | M0 | None | Domain Number | IOP-DP or -TSDP | None | S |
| 1 Batch Default | Period 1 | | | | | | | | S |
| | 2 | | | | | | | | S |
| | 3 | | | | | | | | S |
| | 4 | | | | | | | | S |
| | 5 | | | | | | | | S |
| | 6 | | | | | | | | S |
| | 7 | | | | | | | | S |
| | 8 | | | | | | | | S |
| 2 TSO Default | Period 1 | | | | | | | | S |
| | 2 | | | | | | | | S |
| | 3 | | | | | | | | S |
| | 4 | | | | | | | | S |
| | 5 | | | | | | | | S |
| | 6 | | | | | | | | S |
| | 7 | | | | | | | | S |
| | 8 | | | | | | | | S |
| | Period 1 | | | | | | | | S |
| | 2 | | | | | | | | S |
| | 3 | | | | | | | | S |
| | 4 | | | | | | | | S |
| | 5 | | | | | | | | S |
| | 6 | | | | | | | | S |
| | 7 | | | | | | | | S |
| | 8 | | | | | | | | S |
| | Period 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 ofservice rates sufficiently high to be effective, but they must not be so high that service rates become excessively large. This also results in ineffective control.

Changes to the coefficients may require changes to other parameters in the IPS that are dependent on service value specifications (for example, ASRV, DSRV, 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.

To set the SDCs either:

1. Use the values in the default IPS (IEAIPS00). These should generate reasonable service values. The default coefficients are:
   CPU=10.0
   IOC=5.0
   MSO=3.0
   SRB=10.0
2. 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 tables describe the service consumed per second of execution time by CPU model. The values listed are 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.

*Table 3-2. zSeries 990 2084 Processor Models*

| zSeries 990 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds Task or SRB Execution Time Per Service Unit |
|---|---|---|
| zSeries 990, Model 301 | 21857.9 | 0.000046 |
| zSeries 990, Model 302 | 20752.3 | 0.000048 |
| zSeries 990, Model 303 | 20075.3 | 0.000050 |
| zSeries 990, Model 304 | 19559.9 | 0.000051 |
| zSeries 990, Model 305 | 19047.6 | 0.000053 |
| zSeries 990, Model 306 | 18626.3 | 0.000054 |
| zSeries 990, Model 307 | 18202.5 | 0.000055 |
| zSeries 990, Model 308 | 17777.8 | 0.000056 |
| zSeries 990, Model 309 | 17353.6 | 0.000058 |
| zSeries 990, Model 310 | 17003.2 | 0.000059 |
| zSeries 990, Model 311 | 16666.7 | 0.000060 |

*Table 3-2. zSeries 990 2084 Processor Models (continued)*

| zSeries 990 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds Task or SRB Execution Time Per Service Unit |
|---|---|---|
| zSeries 990, Model 312 | 16326.5 | 0.000061 |
| zSeries 990, Model 313 | 15984.0 | 0.000063 |
| zSeries 990, Model 314 | 15640.3 | 0.000064 |
| zSeries 990, Model 315 | 15296.4 | 0.000065 |
| zSeries 990, Model 316 | 14953.3 | 0.000067 |
| zSeries 990, Model 317 | 14787.4 | 0.000068 |
| zSeries 990, Model 318 | 14611.9 | 0.000068 |
| zSeries 990, Model 319 | 14532.2 | 0.000069 |
| zSeries 990, Model 320 | 14440.4 | 0.000069 |
| zSeries 990, Model 321 | 14349.8 | 0.000070 |
| zSeries 990, Model 322 | 14260.2 | 0.000070 |
| zSeries 990, Model 323 | 14171.8 | 0.000071 |
| zSeries 990, Model 324 | 14084.5 | 0.000071 |
| zSeries 990, Model 325 | 13998.3 | 0.000071 |
| zSeries 990, Model 326 | 13913.0 | 0.000072 |
| zSeries 990, Model 327 | 13828.9 | 0.000072 |
| zSeries 990, Model 328 | 13745.7 | 0.000073 |
| zSeries 990, Model 329 | 13663.5 | 0.000073 |
| zSeries 990, Model 330 | 13582.3 | 0.000074 |
| zSeries 990, Model 331 | 13490.7 | 0.000074 |
| zSeries 990, Model 332 | 13400.3 | 0.000075 |

*Table 3-3. zSeries 900 2064 Processor Models*

| zSeries 900 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds Task or SRB Execution Time Per Service Unit |
|---|---|---|
| zSeries 900, Model 101 | 11585.8 | 0.000086 |
| zSeries 900, Model 102 | 10891.8 | 0.000092 |
| zSeries 900, Model 103 | 10430.2 | 0.000096 |
| zSeries 900, Model 104 | 10081.9 | 0.000099 |
| zSeries 900, Model 105 | 9732.4 | 0.000103 |
| zSeries 900, Model 106 | 9384.2 | 0.000107 |
| zSeries 900, Model 107 | 9153.3 | 0.000109 |
| zSeries 900, Model 108 | 8805.7 | 0.000114 |
| zSeries 900, Model 109 | 8456.7 | 0.000118 |
| zSeries 900, Model 110 | 9334.9 | 0.000107 |
| zSeries 900, Model 111 | 9211.3 | 0.000109 |

*Table 3-3. zSeries 900 2064 Processor Models (continued)*

| zSeries 900 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds Task or SRB Execution Time Per Service Unit |
|---|---|---|
| zSeries 900, Model 112 | 8968.6 | 0.000112 |
| zSeries 900, Model 113 | 8724.1 | 0.000115 |
| zSeries 900, Model 114 | 8602.2 | 0.000116 |
| zSeries 900, Model 115 | 8359.5 | 0.00012 |
| zSeries 900, Model 116 | 8117.7 | 0.000123 |
| zSeries 900, Model 1C1 | 12112 | 0.000083 |
| zSeries 900, Model 1C2 | 11502.5 | 0.000087 |
| zSeries 900, Model 1C3 | 11142.1 | 0.00009 |
| zSeries 900, Model 1C4 | 10781.7 | 0.000093 |
| zSeries 900, Model 1C5 | 10540.2 | 0.000095 |
| zSeries 900, Model 1C6 | 10296 | 0.000097 |
| zSeries 900, Model 1C7 | 10056.6 | 0.000099 |
| zSeries 900, Model 1C8 | 9816 | 0.000102 |
| zSeries 900, Model 1C9 | 9575.1 | 0.000104 |

*Table 3-4. zSeries 800 2066 Processor Models*

| zSeries 800 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds Task or SRB Execution Time Per Service Unit |
|---|---|---|
| zSeries 800, Model 0E1 | 1955.0 | 0.000512 |
| zSeries 800, Model 0A1 | 3907.2 | 0.000256 |
| zSeries 800, Model 0X2 UNI | 4239.5 | 0.000236 |
| zSeries 800, Model 0X2 | 3900.5 | 0.000256 |
| zSeries 800, Model 0B1 | 5612.0 | 0.000178 |
| zSeries 800, Model 0C1 | 6980.8 | 0.000143 |
| zSeries 800, Model 0A2 UNI | 6893.6 | 0.000145 |
| zSeries 800, Model 0A2 | 6341.7 | 0.000158 |
| zSeries 800, Model 001 | 9334.9 | 0.000107 |
| zSeries 800, Model 002 | 8588.3 | 0.000116 |
| zSeries 800, Model 003 | 8121.8 | 0.000123 |
| zSeries 800, Model 004 | 7843.1 | 0.000128 |

*Table 3-5. S/390 9672 Processor Models*

| S/390 9672 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds of Task or SRB Execution Time Per Service Unit |
|---|---|---|
| S/390 9672, Model T16 | 6067.5 | 0.000165 |

*Table 3-5. S/390 9672 Processor Models (continued)*

| S/390 9672 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds of Task or SRB Execution Time Per Service Unit |
|---|---|---|
| S/390 9672, Model T26 | 5643.7 | 0.000177 |
| S/390 9672, Model R36 | 5460.8 | 0.000183 |
| S/390 9672, Model R46 | 5278.8 | 0.000189 |
| S/390 9672, Model R56 | 5158.0 | 0.000194 |
| S/390 9672, Model R66 | 5036.2 | 0.000199 |
| S/390 9672, Model R76 | 4915.5 | 0.000203 |
| S/390 9672, Model R86 | 4733.7 | 0.000211 |
| S/390 9672, Model R96 | 4490.6 | 0.000223 |
| S/390 9672, Model RX6 | 4247.4 | 0.000235 |
| S/390 9672, Model Y16 | 7246.4 | 0.000138 |
| S/390 9672, Model Y26 | 6739.7 | 0.000148 |
| S/390 9672, Model Y36 | 6449.0 | 0.000155 |
| S/390 9672, Model Y46 | 6230.5 | 0.000161 |
| S/390 9672, Model Y56 | 6086.0 | 0.000164 |
| S/390 9672, Model Y66 | 5941.3 | 0.000168 |
| S/390 9672, Model Y76 | 5797.1 | 0.000173 |
| S/390 9672, Model Y86 | 5578.8 | 0.000179 |
| S/390 9672, Model Y96 | 5361.9 | 0.000187 |
| S/390 9672, Model YX6 | 5071.3 | 0.000197 |
| S/390 9672, Model RA6 | 4212.7 | 0.000237 |
| S/390 9672, Model RB6 | 3960.4 | 0.000253 |
| S/390 9672, Model R16 | 5588.5 | 0.000179 |
| S/390 9672, Model R26 | 5141.4 | 0.000194 |
| S/390 9672, Model RC6 | 5029.9 | 0.000199 |
| S/390 9672, Model RD6 | 4918.5 | 0.000203 |
| **S/390 9672 Processor Model** | | |
| S/390 9672, Model X17 | 8346.4 | 0.000120 |
| S/390 9672, Model X27 | 7928.6 | 0.000126 |
| S/390 9672, Model X37 | 7677.5 | 0.000130 |
| S/390 9672, Model X47 | 7511.7 | 0.000133 |
| S/390 9672, Model X57 | 7262.8 | 0.000138 |
| S/390 9672, Model X67 | 7095.3 | 0.000141 |
| S/390 9672, Model X77 | 6762.5 | 0.000148 |

*Table 3-5. S/390 9672 Processor Models (continued)*

| S/390 9672 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds of Task or SRB Execution Time Per Service Unit |
|---|---|---|
| S/390 9672, Model X87 | 6512.0 | 0.000154 |
| S/390 9672, Model X97 | 6177.6 | 0.000162 |
| S/390 9672, Model XX7 | 6010.5 | 0.000166 |
| S/390 9672, Model XY7 | 5759.5 | 0.000174 |
| S/390 9672, Model XZ7 | 5592.5 | 0.000179 |
| S/390 9672, Model Z17 | 9667.7 | 0.000103 |
| S/390 9672, Model Z27 | 9184.8 | 0.000109 |
| S/390 9672, Model Z37 | 8893.8 | 0.000112 |
| S/390 9672, Model Z47 | 8700.4 | 0.000115 |
| S/390 9672, Model Z57 | 8412.2 | 0.000119 |
| S/390 9672, Model Z67 | 8217.8 | 0.000122 |
| S/390 9672, Model Z77 | 7831.6 | 0.000128 |
| S/390 9672, Model Z87 | 7540.1 | 0.000133 |
| S/390 9672, Model Z97 | 7249.7 | 0.000138 |
| S/390 9672, Model ZX7 | 6959.5 | 0.000144 |
| S/390 9672, Model ZY7 | 6765.3 | 0.000148 |
| S/390 9672, Model ZZ7 | 6475.1 | 0.000154 |

*Table 3-6. S/390 3000 Models*

| S/390 3000 Processor Model | Service Units Per Second of Task or SRB Execution Time | Seconds of Task or SRB Execution Time Per Service Unit |
|---|---|---|
| S/390 3000, Model A10 | 1792.1 | 0.000558 |
| S/390 3000, Model A20 | 1582.3 | 0.000632 |

If you plan to use these constants for purposes other than those suggested in this manual, please observe the following limitations:

- Actual customer workloads and performance may vary. For a more exact comparison of processors, see the internal throughput rate (ITR) numbers in *Large Systems Performance Reference (LSPR)*.
- CPU time can vary for different runs of the same job step. One or more of the following factors might cause variations in CPU time: CPU architecture (such as storage buffering), cycle stealing with integrated channels, and the amount of queue searching (see *z/OS MVS System Management Facilities (SMF)*).
- The constants do not account for the effects of PR/SM LPAR mode. For example, a logical 1-way partition in an S/390 9672, Model RX3, has 1090 service units per second, while a 10-way partition on the same machine has 839.3 service units per second.

*Using SMF Task Time:* For installations with no prior service data, the task time reported in SMF record Type 4, 5, 30, 34, and 35 records can be converted to service units using the preceding tables.

***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 also *not* counted for the address space that is the target of the cross memory reference. For a description of SMF EXCP counts, see *z/OS MVS System Management Facilities (SMF)*.

***Main Storage Service:*** A program uses one storage service unit when it holds 50 pages (200K) in central storage frames 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 (central) 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 central storage and their reference patterns 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:

$$\frac{100 \times 10}{50} = 20 \text{ storage service units}$$

**Case 2:**

2 address spaces running concurrently 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:

$$\frac{50 \times 5}{50} = 5 \text{ storage service units}$$

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.

***Weighting Service Components:*** 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.

**Note:** If you are defining your service coefficients for the first time, use an MSO value that is very low (.0001), and CPU=1 and SRB=1.

## 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/E
Special purpose

Each classification is discussed separately. It should be noted that it may be desirable to reclassify address spaces as execution characteristics change. For example, very long (to be defined later) TSO/E 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, 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 require 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 subsystemaddress 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. The MPL adjustment function is not utilized for a subsystem domain with these recommended MPL values.

MPL specifications are not meaningful for domains having only non-swappable address spaces.

***Batch and TSO/E Domains:*** There are crucial differences between the execution characteristics of batch and TSO/E 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/E commands and could therefore overload the system if the number of such jobs allowed in the multi-programming set were not controlled. Such limits, however, are not desirable for most TSO/E 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/E work.

If all TSO/E commands require uniform service, one domain will probably suffice to provide good response time. However, if the TSO/E 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: Basic SRM Parameter Concepts” on page 3-14, the maxMPL value is used tolimit the number of address spaces allowed in a domain in the multiprogramming set while the purpose of the minMPL is to guarantee access to the multiprogramming set for a fixed minimum number of address spaces in a domain.

The minMPL value can be used to implement fixed response requirements. If fast response, for example, is a fixed requirement for short TSO/E 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 the multi-programming set for some minimum number of batch jobs.

For long-running, batch-like TSO/E commands, SRM should be allowed to determine the target MPL in a range of 0-999.

***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 non-privileged job through the RESET command by associating it with a domain having a fixed target MPL of zero, that is, minMPL=maxMPL=0. (A privileged job runs in performance group zero.)

## MPL Target Control

Domain control derived from the target control objectives (ASRV and DSRV) have meaning only for domains whose target MPL is adjustable, and only when there is more than one such domain.

***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.

## Selecting I/O and Dispatching Priorities (Worksheet Number 2)

The following recommendations are made for the selection of dispatching priorities:

- 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/E transactions and long batch jobs, should be assigned to a mean-time-to-wait group to aid throughput.
- Address spaces that are likely to be CPU-bound should be assigned to the lowest mean-time-to-wait group. The mean-time-to-wait algorithm allows SRM to dynamically adjust the dispatching priority.
- To maximize the benefit of the mean-time-to-wait algorithm, work assigned to mean-time-to-wait groups should be assigned to as few groups as possible, ideally just one.
- The IMS control region should be assigned a high dispatching priority, possibly lower than that of VTAM, but higher than the dispatching priorities of initiators and short TSO/E transactions.
- The PVLDP keyword should be used to assign a priority to initiators and privileged programs. You could also control privileged programs with the DP keyword if you explicitly assign them to a performance group other than the subsystem default. The default PVLDP value is the lowest mean-time-to-wait group. The master scheduler's address space is 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.
- 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/E | 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/E 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.

## Performance Group Number Assignment (Worksheet Number 4)

Worksheet number 4 aids in writing an installation control specification (IEAICSxx parmlib member). The worksheet already includes the system-defined subsystems TSO/E and STC. To these, add the installation's job entry subsystem and any interactive system that uses the SYSEVENT macro. 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 similarly named 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/E work, including the TCAM address space
- Individual TSO/E commands
- Groups of commands for an interactive subsystem
- Groups of batch jobs (grouped by class, job name, or userid).

Assign report PGNs where desirable. 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.

## Performance Groups (Worksheet Number 5)

Once performance groups are assigned, decide if the controlover the work in the group should change as the work 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.

**Note:** To conserve space, the worksheet omits the storage isolation parameters and the TSO/E response time parameter. These parameters are used only in exceptional cases.

## Selecting Storage Isolation Values

As described in “Section 3: Advanced SRM Parameter Concepts” on page 3-41, 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. The system considers the working set of an application to include the central storage, expanded storage, and any data spaces, hiperspaces, and VIO in expanded storage that the application uses.

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 allow you the maximum available storage, or specify a maximum working set size larger than the maximum expected number of allocated frames. If you want to limit the working set, specify a maximum working set smaller than the maximum expected number of allocated frames. For example, PWSS=(1000,*) does not exempt the address space from page migration and page stealing as long as the address space has at least 1000 frames.

When specifying the working set size, remember that it describes common and private area pages in processor storage (either central or expanded 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. The page-in rate includes:

- Page faults resolved from auxiliary storage
- Hiperspace reads from auxiliary storage
- Expanded storage hiperspace reads that cannot be satisfied because the page is not in expanded storage.

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 and a maximum working set size. In specifying the minimum working set size, specify a value slightly larger than the 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 (*) for the maximum value, which allows 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 central or expanded storage). Specify a working set size high enough to prevent replacement of too many pages from expanded 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 cannot 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/E 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/E 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/E address spaces to maintain a minimum working set size across swap-outs, and, with sufficient central storage available, you can allow larger swap-in sets in order to reduce demand page-ins for trivial transactions.

## 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/E commands, then medium TSO/E commands, then long TSO/E commands, and to detain batch jobs whenever TSO/E service needs to be increased.

Figure 3-1 shows the IPS that is available as the default via an IEBCOPY during the installation of MVS.

```
/*   THIS IPS PREFERS TSO/E SHORT ABOVE ALL OTHER WORK.  SHORT IS  */
/* DEFINED AS 200 SERVICE UNITS.  MEDIUM TSO/E (DEFINED AS 800     */
/* SERVICE UNITS) IS PREFERRED OVER LONG TSO/E AND BATCH.  LONG    */
/* TSO/E AND BATCH GET EQUAL PREFERENCE. A DELAY DOMAIN IS PROVIDED*/
/* FOR WORK THAT SHOULD BE INHIBITED FROM RUNNING.  DEFAULTS       */
/* ARE TAKEN WHERE APPROPRIATE TO MAKE THE MEMBER MORE READABLE.   */
/*                                                                 */
PVLDP = F54                         /* PRIVILEGED USER             */
IOQ = PRTY                          /*                             */
CPU=10.0,IOC=5.0,MSO=3.0,SRB=10.0   /*                             */
/*                                                                 */
/*                                                                 */
/* ALL DOMAINS EXCEPT THE DELAY DOMAIN HAVE A DEFAULTED MINIMUM    */
/* AND MAXIMUM MPL BUT EXPLICITLY INDICATE THEIR CONTENTION        */
/* INDEX ALGORITHM.                                                */
/*                                                                 */
DMN=1,ASRV=(1000,2000)               /* SHORT TSO/E                */
DMN=2,ASRV=(1000,2000)               /* MEDIUM TSO/E - SEPARATE    */
                                     /*     FOR REPORTING          */
DMN=3,DSRV=(1,999999999              /* LONG TSO/E                 */
DMN=4,ASRV=(500,2000)                /* HOT BATCH                  */
DMN=5,DSRV=(1,999999999)             /* BATCH                      */
DMN=99,CNSTR=(0,0)                   /* DELAY DOMAIN               */
/*                                                                 */
/*                                                                 */
/* DOMAINS PROVIDES ACCESSIBILITY TO MAIN STORAGE.                 */
/* DURATIONS ALLOW CONTROL PARAMETERS TO CHANGE AS TRANSACTIONS AGE.*/
/* IF IT IS DESIRED TO TAKE ADVANTAGE OF THE SEPARATE DOMAIN FOR   */
/* HOT BATCH, WORK MUST BE DEFINED TO PERFORMANCE GROUP 3 VIA THE  */
/* IEAICSXX PARMLIB MEMBER.                                        */
/*                                                                 */
PGN=1, (DMN=5,DP=M2)                 /* BATCH                      */
PGN=2, (DMN=1,DP=F34,DUR=200)        /* TSO/E - SHORT              */
       (DMN=2,DP=F32,DUR=800)        /* ----- - MEDIUM             */
       (DMN=3,DP=M2)                 /* ----- - LONG               */
PGN=3, (DMN=4,DP=F30)                /* HOT BATCH                  */
PGN=99, (DMN=99,DP=M1)               /* DELAY                      */
```

*Figure 3-1. The Default IPS for MVS*

## 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. There are enough initiators.

```
CPU=10.0,IOC=5.0,MSO=3.0,SRB=10.0   /* IPS Example 1               */
TUNIT=3                             /*                            */
PVLDP=F50                           /* INITIATOR DISP.  PRIORITY  */
TSPTRN=(*,3,*)                      /* APL at a low dispatching   */
                                    /* priority 66% OF TIME       */
DMN=1,DSRV=(1,999999999)            /* BATCH EXCEPT CLASS X)      */
DMN=2,ASRV=(100000,200000)          /* BATCH CLASS X              */
PGN=1,(DMN=1,DP=M1)                 /* BATCH EXCEPT CLASS X       */
PGN=2,(DMN=1,DP=F70)                /* IMS                        */
PGN=3,(DMN=1,DP=F0,TSDP=F6,TSGRP=3) /* APL/SV                     */
PGN=4,(DMN=2,DP=M1)                 /* BATCH CLASS X              */
```

*Notes on Example 1:*

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.

**Example 2**

Assume:

- A CICS/TSO/BATCH system with a major emphasis on CICS.
- TSO/E 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=3.0,SRB=10.0
IOQ=PRTY
PVLDP=F5
DMN=1
DMN=2,CNSTR=(2,10),ASRV=(400,800)
DMN=3
DMN=4
PGN=1(DMN=1,DP=M1)                  /* BATCH   */
PGN=2(DMN=2,DP=F9,DUR=150)          /* TSO/E   */
     (DMN=2,DP=F7,DUR=350)
     (DMN=3,DP=M1)
PGN=3,(DP=F8)                       /* JES     */
PGN=4,(DMN=4,DP=F8)                 /* CICS    */
```

*Notes on Example 2:*

- Good CICS response time is dependent on its accessibility to the resource. For this reason, I/O requests are queued by priority and CICS is placed at a fixed dispatching priority equal to JES.
- TSO/E activity is light and the duration for trivial TSO/E is only 150. TSO/E has a higher dispatching priority than CICS to insure that even during heavy CICS activity, trivial TSO/E will be satisfied.
- Service units for JES are accumulated in domain 1. CICS units in domain 4.

## 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 specified response and throughput requirements. The analysis relies on RMF reports, and other measurement data as established by the installation when defining the performance 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/E Response Time is Erratic.

**Detection:**

User feedback or personal experience while using TSO/E 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/E 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/E transactions at a lower dispatching priority than first period TSO/E transactions, and ensure that the long transactions are indeed switching to a new period (DUR value for short TSO/E transactions is not too large).

   Also, examine the use of dispatching priorities higher than or equal to the short TSO/E transaction priority. There may be dispatching queue interference from other work which has a higher dispatching priority.

2. The number of TSO/E 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/E transaction domain. If this value fluctuates, there are several possible solutions:

   a. Increase the target MPL for the short TSO/E transaction domain by raising the minimum MPL above the observed average number of ready 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/E address space becomes ready, it is swapped in. This provides consistent short TSO/E transaction response, but may also mean that short TSO/E transactions consume more system resources than is tolerable, introducing, for example, sudden heavy page demands.

   b. Use the RTO parameter in the first period of the TSO/E performance group, specifying the response time for first period TSO/E transactions. If necessary, the RTO function delays TSO/E commands to achieve the response time objective. The RTO function provides more consistent TSO/E response times during system workload fluctuation. If small, the RTO delay might not be noticed at the terminal.

***Problem 2:*** TSO/E 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/E domains are too heavily favored.

   As TSO/E 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. Use RMF trace to examine the contention index for each domain. If the TSO/E domains are favored heavily, adjust the ASRV/DSRV to obtain the appropriate domain priority scheme.

2. TSO/E has too great a dispatching priority advantage. Possibly, long TSO/E transactions should use the same range of dispatching priorities as batch work. This will help delay long TSO/E response time, but will not slow down short TSO/E response. However, placing any address spaces at an equal or higher priority than short TSO/E transactions may cause erratic response time, and thus is a fairly drastic solution.

***Problem 3:*** TSO/E 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/E 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/E domain, the maximum MPL, and possibly the MPL adjusting parameters should be changed to allow the TSO/E 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/E transactions have a dispatching priority lower than other work, response time may be affected. Ensure that at least long TSO/E transactions are placed at a lower priority than short TSO/E transactions. Refer to the discussion under Problem 1. If there is work that must have a higher dispatching priority than short TSO/E commands, consider using time slicing to lessen the impact of that work on the TSO/E response time.

4. Long-running TSO/E commands may be interfering with short TSO/E response.

   Long TSO/E commands (especially batch-like work) can be detrimental to short TSO/E 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/E domain.

5. Paging overhead may be impacting TSO/E response.

   If RMF paging activity reports indicate a high pageable system area paging rate compared to the address space paging rate, TSO/E 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. Also, reevaluation of IEAPAKxx can help reduce the in-storage size and number of page-ins for the PLPA.

***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, 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.

***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 central 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/E 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.

## 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/E environment, the vast majority of swaps should be of this type. This is not a problem. In a system with available central storage, these should appear as logical rather than physical swaps.

2. "Output terminal wait" swap count is above 0. In a system with available central storage, these should appear as logical rather than physical swaps.

   a. TSO/E 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/E-TCAM, or increasing the HIBFREXT value in the TSOKEY00 parmlib member for TSO/E-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/E 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/E as described under Problem 1 will not include unilateral swaps because the delay is incurred in the first period, before the TSO/E 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. "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 a page rate of more than 20 - 40 per second per local page data set, the installation is probably incurring excessive overhead.

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 central 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 central storage, or decrease the minimum MPL value for at least one domain.

2. 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 (DUR). Also, RMF and SMF transaction and service statistics will differ from previous reports.

3. Not enough initiators have been started.

***Problem 10:*** The System Is Sluggish.

**Detection:**

RMF reports indicate less services than usual accumulated by performance group zero.

*Possible Causes:*

1. The ICS classifies privileged work.

   Prior to MVS/ESA SP 5.1, the privileged indicator in the PPT handled special programs by guaranteeing residency in storage and a high dispatching priority, or you could assign PVLDP in the IPS. Any performance group assignment in the ICS was ignored. With MVS/ESA SP 5.1, you can control privileged work with a non-zero performance group. So, if you have a non-zero performance group assignment in your ICS for MVS/ESA SP 5.1, the performance group assignment is honored. If you do not want to control the privileged work, delete the ICS performance group assignment. For more information, see “Work Specified as Privileged in the PPT or SCHEDxx” on page 3-29.

2. The ICS classifies a special system address space.

3. A workload management policy classifies privileged work.

   If your system is running in goal mode with an active policy that classifies privileged work, keep in mind that once you control privileged work, you control it regardless of the mode in which your system is running. The work does not resume its privileged status when you switch to compatibility mode. If you want the work to regain its privileged status, you must remove any performance group assignment from the ICS, and reIPL.

## Section 5: 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/E session only when any one of the following is true:

- There is no IEAICSxx in effect and, for TSO/E users, the user attribute dataset (UADS) permits the PGN to be assigned.
- The installation control specification omits the applicable subsystem (STC, TSO/E, or job entry subsystem) and, for TSO/E users, the user attribute dataset (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/E, the UADS must permit the PGN.

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/E jobs is 1; for TSO/E 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 on the PERFORM= parameter on the following JES2 initialization statements:
  - JOBCLASS
  - STCCLASS
  - TSUCLASS

  Defaults can be assigned for started tasks, TSO/E users, or any batch class. For more information, see the description of the PERFORM= parameter in *z/OS JES2 Initialization and Tuning Reference*.
- 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/E users, the PERFORM parameter can also be specified on the LOGON command. (See *z/OS TSO/E Command Reference*.) If an IEAICSxx parmlib member is in effect at LOGON, 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/E 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” on page 3-24.

## 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 *z/OS MVS System Commands* .

The RESET command is used to change the control performance group of an executing non-privileged job (a privileged job runs in performance group zero). The new performance group applies to all steps of the job and overrides the performance group assignment in IEAICSxx. An SMF Type 90 Subtype 30 record is written each time a RESET operator command completes successfully. The record identifies the job that was reset, the operator that initiated the command, and the change that was requested.

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. For more information, see "Installation Control Specification Concepts" on page 3-24 and "IPS Concepts" on page 3-15.

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.

# Appendix. Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen-readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

## Using assistive technologies

Assistive technology products, such as screen-readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using it to access z/OS interfaces.

## Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Volume I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

# Notices

This information was developed for products and services offered in the USA.

IBM® may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

## Programming Interface Information

This book is intended to help the customer initialize and tune the MVS element of z/OS. This book documents information that is NOT intended to be used as Programming Interfaces of z/OS.

## Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

- AnyNet
- BookManager
- CICS
- CICS/ESA
- DB2
- DFSMS
- DFSMSdfp
- DFSMSdss
- DFSMShsm
- DFSMSrmm
- DFSMS/MVS
- ES/9000
- ESCON
- GDDM
- Hiperspace
- IBM
- IBMLink
- IMS
- IMS/ESA
- Language Environment
- MVS/ESA
- NetView
- OS/2
- OS/390
- PR/SM

- RACF
- Resource Link
- Resource Measurement Facility
- RETAIN
- RMF
- S/390
- SOM
- SOMobjects
- SP
- SystemView
- VisualLift
- VTAM
- z/OS

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

# Index

## Special characters



## A



## B



## C

## D



## E



## F

# M



# N



# O



# P

# R



# S

# T



# U

# V



# W



# X



# Z

# Readers' Comments — We'd Like to Hear from You

**z/OS**
**MVS Initialization and Tuning Guide**

**Publication No. SA22-7591-02**

**Overall, how satisfied are you with the information in this book?**

| | Very Satisfied | Satisfied | Neutral | Dissatisfied | Very Dissatisfied |
|---|---|---|---|---|---|
| Overall satisfaction | ☐ | ☐ | ☐ | ☐ | ☐ |

**How satisfied are you that the information in this book is:**

| | Very Satisfied | Satisfied | Neutral | Dissatisfied | Very Dissatisfied |
|---|---|---|---|---|---|
| Accurate | ☐ | ☐ | ☐ | ☐ | ☐ |
| Complete | ☐ | ☐ | ☐ | ☐ | ☐ |
| Easy to find | ☐ | ☐ | ☐ | ☐ | ☐ |
| Easy to understand | ☐ | ☐ | ☐ | ☐ | ☐ |
| Well organized | ☐ | ☐ | ☐ | ☐ | ☐ |
| Applicable to your tasks | ☐ | ☐ | ☐ | ☐ | ☐ |

**Please tell us how we can improve this book:**

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

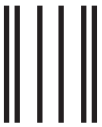
Company or Organization _____

Phone No. _____

**Readers' Comments — We'd Like to Hear from You**
SA22-7591-02

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 MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY
12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape



Cut or Fold
Along Line

Program Number: 5694-A01

Printed in USA

SA22-7591-02