MVS

Programming:
Batch Local Shared Resources Subsystem Guide

GC28-1469-00

MVS

Programming:
Batch Local Shared Resources Subsystem Guide

GC28-1469-00

Note

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

First Edition, June 1994

This edition applies to Version 5 Release 1 of MVS/ESA System Product (5655-068 or 5655-069) and to all subsequent versions, releases, and modifications until otherwise indicated in new editions or technical newsletters.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

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

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

FAX (United States & Canada): 914+432-9405
FAX (Other Countries):
Your International Access Code +1+914+432-9405

IBMLink (United States customers only): KGNVMC(D58PUBS)
IBM Mail Exchange: USIB2NZL at IBMMAIL
Internet: d58pubs@vnet.ibm.com

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 book
- 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

Notices

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates.

Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, programs, or services except those expressly designated by IBM, are the user's responsibility.

IBM may have patents or pending patent applications covering subject matter 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
500 Columbus Avenue
Thornwood, New York 10594
USA

Programming Interfaces

This book is intended to help customers to enable applications to use VSAM local shared resources (LSR). It contains procedures for:

- installing the batch LSR subsystem
- converting existing applications to use batch LSR

This book primarily documents General-Use Programming Interface and Associated Guidance Information provided by MVS/ESA System Product, as of Version 5.

General-Use programming interfaces allow the customer to write programs that request or receive the services of MVS/ESA System Product, as of Version 5.

However, this book also documents Product-Sensitive Programming Interface and Associated Guidance Information.

Product-Sensitive programming interfaces are provided to allow the customer installation to perform tasks such as tailoring, monitoring, modification, or diagnosis of this IBM product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM product. Product-Sensitive interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-Sensitive programming interface information is explicitly identified where it occurs, either as an introductory statement to a chapter or section that is entirely product-sensitive programming interface information, or is marked as follows:

Product-Sensitive Programming Interface

Product-Sensitive Programming Interface and Associated Guidance Information

End of Product-Sensitive Programming Interface

Trademarks

The following terms, **DENOTED BY AN ASTERISK (*)** on first use in the text of this publication, are trademarks of the IBM Corporation in the United States and/or other countries:

- DFSMS/MVS
- Hiperbatch
- Hiperspace
- IBM
- IBMLink
- MVS/DFP
- MVS/ESA
- RACF

About This Book

This book explains how to enable applications to access virtual storage access method (VSAM) data sets as if they are VSAM local shared resources (LSR). Without the batch LSR subsystem, applications normally access VSAM data sets as VSAM non-shared resources (NSR). With the batch LSR subsystem, you can improve performance of certain applications. This book describes how to install and use the batch local shared resources (LSR) subsystem.

Who Should Use This Book

This book is written for those using the batch local shared resources (LSR) subsystem including:

- Those responsible for installing and maintaining MVS/ESA.
- System programmers responsible for the performance of applications using VSAM.

It is assumed that the reader is knowledgeable about job control language (JCL) and VSAM concepts and terminology.

How to Use This Book

This book is one of the set of programming books for MVS. This set describes how to write programs in assembler language or high-level languages, such as C, FORTRAN, and COBOL. The following tables show how this book fits in with the others in the set:

Note: The book titles and order numbers listed in the following tables pertain to MVS/ESA SP5.1.0. If you are using a different release of MVS, please refer to the corresponding library guide to find the correct book titles and order numbers for your release.

Programming Books for Assembler Language Programs

Book Title	Use this book to:	Order Number
MVS/ESA SP V5 Assembler Services Guide	Find out how to use system services provided by macros and callable services available to all assembler language programs. If you are relatively new to assembler language programming, this book is a good place to start.	GC28-1466
MVS/ESA SP V5 Assembler Services Reference	Learn how to code macros and callable services that are available to all assembler language programs. This book is for all assembler language programmers.	GC28-1474
MVS/ESA SP V5 Auth Assembler Services Guide	Find out how to use system services provided by macros and callable services that are available to programs running in supervisor state or with PSW key 0-7 or that are APF-authorized programs. This book is for experienced assembler language programmers; it assumes, for example, that you are familiar with the information in *MVS/ESA SP V5 Assembler Services Guide*.	GC28-1467

Book Title	Use this book to:	Order Number
MVS/ESA SP V5 Auth Assembler Services Reference ALE-DYN *MVS/ESA SP V5 Auth Assembler Services Reference ENF-ITT* *MVS/ESA SP V5 Auth Assembler Services Reference LLA-SDU* *MVS/ESA SP V5 Auth Assembler Services Reference SET-WTO*	Learn how to code macros and callable services that are available to programs running in supervisor state or with PSW key 0-7 or that are APF-authorized programs. This book is for experienced assembler language programmers.	GC28-1475 GC28-1476 GC28-1477 GC28-1478
MVS/ESA SP V5 Extended Addressability Guide	Find out how to extend or restrict the storage available to programs through the use of access registers, cross memory services, data spaces, subspaces, and hiperspaces. This book is for experienced assembler language programmers.	GC28-1468
MVS/ESA SP V5 JES Common Coupling Services	Learn how to use the JES common coupling services and exits to affect JES communication processing in an MVS sysplex environment. This book is for experienced JES2, JES3, and BCP assembler language system programmers.	GC28-1497
MVS/ESA SP V5 Sysplex Services Guide	Find out how to use authorized system services to enable your multisystem application or subsystem to run in a sysplex, benefit from automatic restarts, and share data using the coupling facility. This book is for experienced assembler language programmers.	GC28-1495
MVS/ESA SP V5 Sysplex Services Reference	Learn how to code authorized system services to enable your multisystem application or subsystem to run in a sysplex, benefit from automatic restarts, and share data using the coupling facility. This book is for experienced assembler language programmers.	GC28-1496
MVS/ESA SP V5 Workload Management Services	Find out how to use workload management services that are intended for subsystem work manager and performance monitoring programs.	GC28-1494

Book Title	Use this book to:	Order Number
MVS Batch Local Shared Resources	Find out whether your installation's batch applications can benefit from batch LSR subsystem and how to use batch LSR subsystem. Using this book does not require a knowledge of assembler language programming.	GC28-1469
MVS Hiperbatch Guide	Find out whether your installation's batch applications can benefit from Hiperbatch and how to use Hiperbatch. Using this book does not require a knowledge of assembler language programming.	GC28-1470

Programming Books for High-Level Language Programs

Book Title	Use this book to:	Order Number
MVS/ESA SP V5 Callable Services for HLL	Find out how to use and code program CALLs for specific MVS services in high-level language (HLL) programs.	GC28-1464

Programming Books for Both Assembler and High-Level Language Programs

Book Title	Use this book to:	Order Number
&ieaq200t.	Learn how to use registration services to register a product or to prepare an optional OS/390 element dynamic enablement. This book is for assembler or C language programmers.	&ieaq200n.
MVS/ESA SP V5 Writing TPs for APPC/MVS	Find out how to use and code program CALLs for APPC/MVS communication services that are intended for both APPC/MVS transaction programs and servers. This book is a companion to *MVS/ESA SP V5 Writing Servers for APPC/MVS*, which documents APPC/MVS services for servers only.	GC28-1471
MVS/ESA SP V5 Writing Servers for APPC/MVS	Find out how to use and code program CALLs for APPC/MVS communication services that are intended for servers only. This book is a companion to *MVS/ESA SP V5 Writing TPs for APPC/MVS*, which documents the APPC/MVS services for both transaction programs and servers.	GC28-1472
MVS/ESA SP V5 Writing Transaction Schedulers for APPC/MVS	Find out how to use and code program CALLs for APPC/MVS system services that are intended for APPC/MVS transaction schedulers. This book is for programmers who write transaction schedulers other than the one APPC/MVS provides.	GC28-1465

Where to Find More Information

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

Note: The book titles and order numbers listed in the table pertain to MVS/ESA SP5.1.0. If you are using a different release of MVS, please refer to the corresponding library guide to find the correct book titles and order numbers for your release.

Short Title Used in This Book	Title	Order Number
MVS/ESA SP V5 Initialization and Tuning Guide	*MVS/ESA Initialization and Tuning Guide*	SC28-1451
MVS/ESA SP V5 JCL Reference	*MVS/ESA JCL Reference*	GC28-1479
MVS/ESA SP V5 JES Common Coupling Services	*MVS/ESA Programming: JES Common Coupling Services*	GC28-1497
MVS/ESA SP V5 Workload Management Services	*MVS/ESA Programming: Workload Management Services*	GC28-1494
MVS/DFP Using Data Sets	*MVS/DFP V3 Using Data Sets*	SC26-4749
	DFSMS/MVS Version 1 Release 1 Using Data Sets	SC26-4922
MVS/DFP Macro Instructions for Data Sets	*MVS/DFP V3 Macro Instructions for Data Sets*	SC26-4747
	DFSMS/MVS Version 1 Release 1 Macro Instructions for Data Sets	SC26-4913
RACF V2 Security Administrator's Guide[1]	*RACF V2 Security Administrator's Guide*	SC23-3726
	RACF V1 R9.2 Security Administrator's Guide	SC28-1340
MVS/ESA SP V5 System Management Facilities (SMF)	*MVS/ESA System Management Facilities (SMF)*	GC28-1457
MVS/ESA SP V5 Routing and Descriptor Codes	*MVS/ESA Routing and Descriptor Codes*	GC28-1487
MVS/ESA SP V5 System Messages, Vol 1 (ABA-ASA)	*MVS/ESA System Messages, Volume 1 (ABA-ASA)*	GC28-1480
MVS/ESA SP V5 System Messages, Vol 2 (ASB-EWX)	*MVS/ESA System Messages, Volume 2 (ASB-EWX)*	GC28-1481
MVS/ESA SP V5 System Messages, Vol 3 (GDE-IEB)	*MVS/ESA System Messages, Volume 3 (GDE-IEB)*	GC28-1482
MVS/ESA SP V5 System Messages, Vol 4 (IEC-IFD)	*MVS/ESA System Messages, Volume 4 (IEC-IFD)*	GC28-1483
MVS/ESA SP V5 System Messages, Vol 5 (IGD-IZP)	*MVS/ESA System Messages, Volume 5 (IGD-IZP)*	GC28-1484
MVS/ESA SP V5 Assembler Services Guide	*MVS/ESA Programming: Assembler Services Guide*	GC28-1466
MVS/ESA SP V5 Assembler Services Reference	*MVS/ESA Programming: Assembler Services Reference*	GC28-1474
MVS/ESA SP V5 Auth Assembler Services Guide	*MVS/ESA Programming: Authorized Assembler Services Guide*	GC28-1467

Short Title Used in This Book	Title	Order Number
MVS/ESA SP V5 Auth Assembler Services Reference ALE-DYN	*MVS/ESA Programming: Authorized Assembler Services Reference, Volume 1 (ALESERV-DYNALLOC)*	GC28-1475
MVS/ESA SP V5 Auth Assembler Services Reference ENF-ITT	*MVS/ESA Programming: Authorized Assembler Services Reference, Volume 2 (ENFREQ-ITTFMTB)*	GC28-1476
MVS/ESA SP V5 Auth Assembler Services Reference LLA-SDU	*MVS/ESA Programming: Authorized Assembler Services Reference, Volume 3 (LLACOPY-SDUMPX)*	GC28-1477
MVS/ESA SP V5 Auth Assembler Services Reference SET-WTO	*MVS/ESA Programming: Authorized Assembler Services Reference, Volume 4 (SETFRR-WTOR)*	GC28-1478
MVS/ESA SP V5 Extended Addressability Guide	*MVS/ESA Programming: Extended Addressability Guide*	GC28-1468
MVS Hiperbatch Guide	*MVS Programming: Hiperbatch Guide*	GC28-1470
MVS Batch Local Shared Resources	*MVS Programming: Batch Local Shared Resources Subsystem Guide*	GC28-1469
MVS/ESA SP V5 Callable Services for HLL	*MVS/ESA Programming: Callable Services for High-Level Languages*	GC28-1464
MVS/ESA SP V5 Writing Transaction Schedulers for APPC/MVS	*MVS/ESA Programming: Writing Transaction Schedulers for APPC/MVS*	GC28-1465
MVS/ESA SP V5 Writing TPs for APPC/MVS	*MVS/ESA Programming: Writing Transaction Programs for APPC/MVS*	GC28-1471
MVS/ESA SP V5 Writing Servers for APPC/MVS	*MVS/ESA Programming: Writing Servers for APPC/MVS*	GC28-1472
MVS/ESA SP V5 Sysplex Services Guide	*MVS/ESA Programming: Sysplex Services Guide*	GC28-1495
MVS/ESA SP V5 Sysplex Services Reference	*MVS/ESA Programming: Sysplex Services Reference*	GC28-1496

[1] Select the book that supports the version of the product your installation is using.

Summary of Changes

Summary of Changes
for GC28-1469-00
as Updated June, 1996
online only for SK2T-0710-15

The following changes appear only in the online version of this publication. A pair of vertical dots (:) in the left margin indicates changes to the text and illustrations.

This revision reflects the deletion, addition, or modification of information to support miscellaneous maintenance items.

Summary of Changes
for GC28-1469-00
as Updated June, 1994.

This book contains information previously presented in *MVS/ESA Application Development Guide: Batch Local Shared Resources Subsystem*, GC28-1672-01, which supports MVS/ESA System Product Version 4 Release 3.

Chapter 1. Overview of the Batch Local Shared Resources (LSR) Subsystem

Using the batch local shared resources (LSR) subsystem provides advantages in a virtual storage access method (VSAM) application:

- Improved performance when the same control interval (CI) is referenced more than once in the processing, but other processing intervenes between the two references.
- Ability to use a single set of buffers of a particular size for multiple cluster components having the same, or smaller, CI size.
- Ability to force VSAM buffers and control blocks to be located above 16 megabytes (Mb) without having to use hiperspaces.

The batch LSR subsystem improves performance by allowing programs that use VSAM non-shared resources (NSR) to switch to use VSAM local shared resources (LSR) without changing the application source code or link-editing the application again. Only a JCL change is required.

If the restrictions documented in this publication are met, batch LSR can be used by programs coded in high-level languages as well as assembler language programs.

The VSAM data set types supported are:

- Key sequenced data set (KSDS)
- Relative record data set (RRDS), including variable RRDS (VRRDS)
- Entry sequenced data set (ESDS).

Because JCL can specify either a path name or a cluster name, environments with alternate indexes are also supported. Batch LSR can also be used with SMS-managed data sets.

Programs that use batch LSR can use a hiperspace* or deferred write. The installation can control the ability to create hiperspaces batch LSR by use of a RACF* facility class, if desired.

For more information about using shared resources, including LSR, see the following manuals:

- *MVS/DFP Using Data Sets*
- *MVS/DFP Macro Instructions for Data Sets*

Virtual Storage Access Method (VSAM) Buffer Management

To understand the advantages of using the batch LSR subsystem for your VSAM processing, it is helpful to understand the differences in the way VSAM manages buffers for NSR and LSR. A VSAM GET or PUT request is processed the same way whether the request is for NSR or LSR, except for buffer management processing. At the buffer management level, where VSAM manipulates, fills, and writes out buffer contents, processing differs depending on whether the user specifies NSR (the default) or LSR in the MACRF parameter on the ACB macro:

- For NSR, data buffers belong to a particular string (loosely, a request parameter list (RPL)). When an application uses an RPL to issue a direct GET for a record with a key of 1234 using NSR, VSAM manages buffers as follows:

  1. VSAM locates the correct CI.
  2. VSAM then reads the data CI into storage and gives the user the requested record.
  3. If the user then issues another direct GET for a record with a key of 6789, which is in a different CI from record 1234, the same data buffer gets used for this request and the CI containing 6789 overlays the CI containing 1234.
  4. If the user then issues another direct GET for a record with key 1235 (which is in the same CI as 1234), that CI must be read in again because the intervening GET for 6789 causes the previous buffer contents to be lost.

  This problem cannot be solved by using multiple strings. If, for example, the requests for 1234, 6789, and 1235 use three different RPLs, both CIs will be in storage when the request for 1235 occurs, but VSAM will not look in another string's buffers to satisfy the request. The CI will be read in again anyway.

  These NSR characteristics lead to performance complaints in cases where the processing is random but the same CI is requested multiple times with intervening requests to other CIs.

- For LSR, data buffers are managed so that buffers will not be overlaid:

  1. The user builds a resource pool using the BLDVRP macro, and issues the OPEN macro specifying LSR on the MACRF parameter of the ACB to open the cluster.
  2. If the user requests record 1234, for example, it is brought into storage.
  3. If the user then requests record 6789, for example, that record will be brought into storage and placed into a buffer as follows:
     - As long as sufficient buffers are available, record 6789 will be read into a different buffer than record 1234. If sufficient buffers are available, a subsequent request for record 1235 can be satisfied by using the CI that was brought into storage to satisfy the request for 1234, even if the same RPL is used over and over again.
     - If sufficient buffers are not available, the “least recently used” CI's buffer will be overlaid to satisfy the new request.

Identifying Jobs that Might Benefit from Batch LSR

The jobs that might benefit from batch LSR are those with certain application characteristics (such as data reference patterns), and data characteristics (such as the relationship between record length and data control interval size). The best candidates are:

- Long-running, direct processing jobs
- Jobs with a high execute channel program (EXCP) count.

Note: You should not consider job steps that are totally sequential because the run times for such steps could increase considerably using batch LSR.

To identify possible candidates for batch LSR you need one of the following versions of DFP installed on the system used to gather the SMF data:

- DFP Version 2.3 with APAR OY23661
- DFP Version 3.1 or above with APAR OY20341

To determine candidates for batch LSR, use SMF type 64 records to obtain information about the batch LSR candidate's processing characteristics, including:

- Jobname
- Cluster name
- Component identifier (data or index)
- Component name
- Change in number of EXCPs
- ACB MACRF fields.

For instance:

- SMF64MC1 indicates whether a job uses direct processing, sequential processing, or a mixture of the two. Although jobs using direct processing are the most likely prospects, jobs using mixed processing are also worth evaluating as possible choices.
- SMF64DEP indicates EXCP count changes. Because batch LSR avoids I/O by using virtual storage, you must evaluate the cost versus benefit of the trade-off.

You cannot, however, determine from the type 64 record whether the job reaccesses data in the same control interval. See “Assessing and Tuning With Batch LSR” on page 3-14 for information about how to determine whether the candidates for batch LSR show performance improvement when run with batch LSR.

Chapter 2. Installing Batch LSR

Step 1. Add the following line to member IEFSSNxx of SYS1.PARMLIB:

```
ssnm,CSRBISUB
```

Where *ssnm* is the name of the batch LSR subsystem. BLSR is the suggested name.

Member IEFSSNxx is identified on the SSN parameter in the IEASYSxx member you use when you initial program load (IPL) your system (response to message IEA101A).

: Step 2. Re-IPL the system, or issue the following MVS command:

:
```
SETSSI ADD,SUBNAME=ssnm,INITRTN=CSRBISUB
```

If the installation is successful, message CSR002I appears on the operator console, indicating that batch LSR initialization is complete.

Step 3. As a default, the batch LSR subsystem allows all users to create a VSAM hiperspace using the HBUFND and HBUFNI subparameters of the batch LSR subsystem SUBSYS parameter. If you want to restrict this capability to a selected set of users, you can:

Step a. Issue the RACF SETROPTS command with the CLASSACT operand to activate the FACILITY class, if this class is not already active. The format of the command is:

```
SETROPTS CLASSACT(FACILITY)
```

Step b. Issue the RACF RDEFINE command to set the default access for the resource “CSR.BLSRHIPR.ssnm” to none:

```
RDEFINE FACILITY CSR.BLSRHIPR.ssnm UACC(NONE)
```

Note that *ssnm* is the name of the batch LSR subsystem. BLSR is the suggested name.

Step c. Define users authorized to create a VSAM hiperspace using the batch LSR subsystem. For example:

```
PERMIT 'CSR.BLSRHIPR.BLSR' CLASS(FACILITY) ID(usera) ACCESS(READ)
```

For more information on the FACILITY class, see *RACF V2 Security Administrator's Guide*.

Chapter 3. Using the Batch LSR Subsystem

To convert VSAM NSR processing to LSR processing:

- Use the SUBSYS parameter in the JCL. Follow the example in "SUBSYS Parameter" below.
- Install the batch LSR subsystem following the procedure in Chapter 2, "Installing Batch LSR" on page 2-1.

The application program does not have to be changed or link-edited again.

Note: You will receive a JCL error if you run a job specifying the use of batch LSR on a system that does not support the batch LSR subsystem.

The batch LSR subsystem supports:

- Specification of either a cluster name or a path name in the JCL. The data set may be either a keyed-sequential data set (KSDS), a relative record data set (RRDS), or an entry sequenced data set (ESDS).
- Determination of the maximum data or index CI size in the "aggregate," including the cluster, alternate index, and all members of an upgrade set.

SUBSYS Parameter

To convert to LSR processing, add the SUBSYS parameter to the JCL. The following example illustrates how to do this.

If, for example, the application opens the following data set for NSR processing:

```
//VSAMDD  DD DISP=SHR,DSN=VSAMDSN
```

Convert to the batch LSR subsystem as follows:

Step 1. Change the DDNAME on the previous JCL command statement from VSAMDD to VSAMALT:

```
//VSAMALT DD DISP=SHR,DSN=VSAMDSN
```

Step 2. Add the following DD statement, where the SUBSYS subsystem-name subparameter is BLSR and the SUBSYS DDNAME subparameter is the DD name selected in step 1:

```
//VSAMDD  DD SUBSYS=(BLSR,'DDNAME=VSAMALT')
```

When the application opens the VSAMDD access method control block (ACB), the batch LSR subsystem completes the conversion to VSAM LSR processing. The following system parameters are not allowed with the SUBSYS parameter: *, AMP, BURST, CHARS, COPIES, DATA, DDNAME, DYNAM, FLASH, MODIFY, QNAME, SPACE, SYSOUT. You must nullify any of these parameters if they are specified on a DD statement you are overriding.

SUBSYS Subparameters

The subsystem-name and DDNAME are required subparameters on the SUBSYS parameter. However, other subparameters may be specified on the SUBSYS parameter:

```
//VSAMDD DD SUBSYS=(subsystem-name,'DDNAME=value',
//          'subparm1=value',....,'subparmn=value')
```

The SUBSYS subparameters can be specified in either of two ways:

```
subparm=value
subparm(value)
```

and must be separated by either commas, blank spaces, or both. The following are equivalent:

```
subparm1=value1,subparm2(value2)
subparm1(value1) subparm2=value2
subparm1(value1) , subparm2=value2
subparm1(value1) , , subparm2=value2
```

If you code a subparameter without specifying a value, for instance:

```
subparm1=,
subparm1()
```

batch LSR takes the default value for the subparameter. If there is no default value, batch LSR ignores the subparameter.

Note: Although the JCL SUBSYS parameter cannot be used with SMS-managed data sets, there is no such restriction on the batch LSR SUBSYS parameter. Batch LSR **can** be used with SMS-managed data sets.

The following subparameters are allowed on the batch LSR SUBSYS parameter:

Subparameter	Meaning and Range
subsystem-name	The name of the subsystem. The subsystem-name is 1 through 4 alphanumeric or national ($,#,@) characters; the first character must be alphabetic or national ($,#,@). The subsystem must be available in the installation. This subparameter is required.
DDNAME	The name of the JCL command statement that points to the VSAM data set. This subparameter is required. **Note:** DDNAME should not be confused with the system parameter, DDNAME.
BUFND	The number of virtual buffers to create in the data buffer pool. If not specified, it defaults to the: • Number of buffers that fit in 5 megabytes if the buffers are acquired above 16 megabytes. • Number of buffers that fit in 250 kilobytes if the buffers are acquired below 16 megabytes. The value can range from 10 to 32000. **Note:** The default number of data and index buffers will be created even if BUFND and BUFNI are omitted and HBUFND and HBUFNI are specified.

For both BUFND and BUFNI, the decision of whether to acquire buffers above or below 16 megabytes is based on the value specified for the RMODE31 SUBSYS subparameter, if any. If the RMODE31 SUBSYS subparameter is not specified, the residency location defaults to the RMODE31 specification on the MACRF parameter of the user's ACB. If RMODE31=ALL or RMODE31=BUFF is specified in either the SUBSYS subparameter or the user's ACB, buffers are acquired above 16 megabytes.

BUFNI The number of virtual buffers to create in the index buffer pool. If not specified, it defaults to the:

- Number of buffers that fit in 5 megabytes if the buffers are acquired above 16 megabytes.
- Number of buffers that fit in 250 kilobytes if the buffers are acquired below 16 megabytes.

The value can range from 10 to 32000.

For both BUFND and BUFNI, the decision of whether to acquire buffers above or below 16 megabytes is based on the value specified for the RMODE31 SUBSYS subparameter, if any. If the RMODE31 SUBSYS subparameter is not specified, the residency location defaults to the RMODE31 specification in the MACRF parameter of the user's ACB. If RMODE31=ALL or RMODE31=BUFF is specified in either the SUBSYS subparameter or the user's ACB, buffers are acquired above 16 megabytes.

HBUFND The number of hiperspace buffers in the data buffer pool. If not specified, a hiperspace data buffer pool is not created.

Hiperspaces are created as follows:

- If the Resource Access Control Facility (RACF) resource CSR.BLSRHIPR.*subsystem-name* in the FACILITY class **is not** defined, a hiperspace **is** created.
- If the RACF resource CSR.BLSRHIPR.*subsystem-name* in the FACILITY class **is** defined, a hiperspace **will not** be created, unless the user is also authorized to use this resource.

If the user cannot create a hiperspace, LSR processing is still used. Only the buffer pool in the address space is created. If a hiperspace is used, the size of a data buffer is rounded up to a multiple of 4K.

The value can range from 5 to 32000.

HBUFNI The number of hiperspace buffers in the index buffer pool. If not specified, a hiperspace index buffer pool is not created.

Hiperspaces are created as follows:

- If the RACF resource CSR.BLSRHIPR.*subsystem-name* in the FACILITY class **is not** defined, a hiperspace **is** created.

- If the RACF resource CSR.BLSRHIPR.*subsystem-name* in the FACILITY class **is** defined, a hiperspace **will not** be created, unless the user is also authorized to use this resource.

If the user cannot create a hiperspace, LSR processing is still used. Only the buffer pool in the address space is created. If a hiperspace is used, the size of an index buffer is rounded up to a multiple of 4K.

The value can range from 5 to 32000.

RMODE31 Specifies whether control blocks and/or buffers associated with the LSR pool will be located above 16 megabytes. Values specified here override the equivalent parameters in the user's ACB and the batch LSR defaults. Allowable values are:

ALL Buffers and control blocks are to be located above 16 megabytes.

BUFF Buffers but not control blocks are to located above 16 megabytes.

CB Control blocks but not buffers are to be located above 16 megabytes.

NONE Neither buffers nor control blocks are to be located above 16 megabytes. Both will be placed below 16 megabytes.

If RMODE31 is not specified, control blocks will be placed above 16 megabytes by default and buffers will be placed as specified by the RMODE31 subparameter of the MACRF parameter of the user's ACB. See “Controlling Buffer and Control Block Residency” on page 3-7 for a detailed explanation of buffer and control block residency considerations and implications.

STRNO Indicates the number of concurrent strings to be used when establishing the LSR pool. The allowable values range from 1 to 255 using a maximum of three digits. The default value is 16.

IBM* recommends that you use the default number of strings. If the number of strings is insufficient, you will receive a return code of 8 in register 15 with a reason code of 64 (X'40') in the feedback word of the request parameter list (RPL) on your VSAM operation. See *MVS/DFP Macro Instructions for Data Sets* for more information. Programs coded in high-level languages generally only require a STRNO value of 1. However, a larger number might be needed if you are using alternate indexes. Using a STRNO value of 16 only requires 4K of storage, so there is little value in specifying a STRNO value smaller than 16. If you are using alternate indexes, you can calculate an appropriate STRNO value by adding the number of concurrent strings needed to the number of alternate indexes being used.

Some assembler language programs may have STRNO requirements that vary according to the path executed or the environment in which the processing is taking place. In this situation, the program will work on some occasions and fail on others when the same STRNO value is specified. For more information about the STRNO parameter, refer to the ACB and BLDVRP macros in *MVS/DFP Macro Instructions for Data Sets*.

DEFERW Indicates whether VSAM deferred write (DFR) is to be used. With non-deferred write (NDF), VSAM immediately writes a CI, for which a direct PUT is issued, to DASD. With DFR and shared resources, VSAM does not do the write until the buffer occupied by the CI is needed for another request. The benefit of DFR is that it can avoid I/O, particularly in cases where there are multiple updates to the same CI (multiple PUTs that resolve to the same CI). The use of DFR, however, may be prohibitive in certain environments. See "VSAM Deferred Write (DFR)" on page 3-8 for more information about deferred write.

Note: If you specify SHAREOPTIONS 4 with DEFERW=YES on a DEFINE or ALTER IDCAMS statement for the cluster, the data set will be processed using batch LSR but not using DFR. If you have specified MSG=I as a subparameter on the SUBSYS parameter, you will also receive message CSR008I stating that deferred write is not supported with SHAREOPTIONS 4. See *MVS/ESA SP V5 System Messages, Vol 2 (ASB-EWX)* for more information about this message.

Allowable values are:

YES Use DFR
NO Do not use DFR

The default is to use the value specified in the ACB (which is usually non-deferred write).

SHRPOOL The identifier of the LSR pool to be used. Pool identifiers range from 0 to 15. If you do not specify an identifier, the system attempts to assign an unused pool for the request. If none are available, you will receive message CSR023I and LSR processing will not be used.

Note: The system considers a pool to be available if both of the following are true:

- The DATA buffer pool associated with the chosen pool identifier is not already in use
- The INDEX buffer pool (if there is one) associated with the chosen pool identifier is not already in use.

If the LSR pool you specify already exists at the time of the OPEN, you will receive message, CSR024I, stating that you have specified an existing LSR pool and that batch LSR will use it. The OPEN request will fail, however, if the characteristics of the specified LSR pool, such as buffer size, are not compatible with those of your request.

BUFSD The size of the data buffer to acquire. If not specified, the size of the data set's data component is used.

The value can range from 1 to 32768.

BUFSI The size of the index buffer to acquire. If not specified, the size of the data set's index component is used.

The value can range from 1 to 32768.

MSG Indicates if informational, warning, or error messages are placed in the user's job log. Allowable values are:

E Only error messages
W Warning and error messages
I Informational, warning and error messages

The default is “W”.

Examples of the SUBSYS Parameter

Several examples of the SUBSYS parameter follow:

```
//VSAMDD  DD   SUBSYS=(BLSR,'DDNAME=VSAMDD2')
```

This is the simplest example, where all defaults are taken. In most cases, this statement is sufficient.

```
//VSAMDD  DD   SUBSYS=(BLSR,'DDNAME=VSAMDD2',
//             'BUFNI=100,BUFND=100,HBUFND=1000')
```

This example illustrates allocating a small index pool and a large data pool. Because this is a large 24-bit program, the user wants a hiperspace to increase the size of the data pool.

```
//VSAMDD  DD   SUBSYS=(BLSR,'DDNAME=VSAMDD2',
//             'SHRPOOL=2')
//VSAMDDA DD   SUBSYS=(BLSR,'DDNAME=VSAMDD3',
//             'SHRPOOL=2')
```

This example illustrates defining a pool shared by 2 OPEN requests. These could be 2 data sets that are concurrently opened or serially opened. By specifying SHRPOOL=*value*, only a single pool is built and assigned to both of the DD statements.

Controlling Buffer and Control Block Residency

The RMODE31 SUBSYS subparameter provides a means of specifying whether VSAM control blocks and buffers are to be placed below or above 16 megabytes. If the RMODE31 SUBSYS subparameter is omitted, the batch LSR default values and the user's ACB values will determine the location of the VSAM buffers and control blocks. The following figure shows the RMODE31 value resulting from the interaction between the RMODE31 SUBSYS subparameter value and the RMODE31 MACRF subparameter value:

RMODE31 SUBSYS Subparameter

RMODE31 MACRF Parameter on ACB	**Omitted**	**NONE**	**BUFF**	**CB**	**ALL**
NONE	CB	NONE	BUFF	CB	ALL
BUFF	ALL	NONE	BUFF	CB	ALL
CB	CB	NONE	BUFF	CB	ALL
ALL	ALL	NONE	BUFF	CB	ALL

Figure 3-1. RMODE31 Values Used by Batch LSR

Message CSR022I is issued to show the RMODE31 values specified with the ACB and with the SUBSYS subparameter. The RMODE31 value resulting from the interaction of the two specifications is used to create the VSAM buffer pool when a data set OPEN is processed. See *MVS/ESA SP V5 System Messages, Vol 2 (ASB-EWX)* for more information about this message. You will only see this message if you have specified MSG=I as a SUBSYS subparameter.

User programs should have no dependencies on VSAM control blocks residing below 16 megabytes. If such a dependency exists, the most likely symptom will be an 0C4 abend. If possible, eliminate VSAM control block dependencies. If you cannot change the code to eliminate these dependencies, you can specify RMODE31=BUFF or RMODE31=NONE on the RMODE31 SUBSYS subparameter to force control blocks or both control blocks and buffers below 16 megabytes. This specification overrides the batch LSR default control block location, which is above 16 megabytes.

If a program has a dependency on VSAM buffers being located below 16 megabytes, you can force the creation of VSAM buffers or VSAM buffers and control blocks below 16 megabytes by specifying RMODE31=CB or RMODE31=NONE on the RMODE31 SUBSYS subparameter or the RMODE31 MACRF subparameter on the user's ACB. This specification overrides the batch LSR default buffer residency location, which is the location specified by the RMODE31 MACRF subparameter in the ACB.

VSAM Deferred Write (DFR)

Deferred write (DFR) is a VSAM LSR option which is specified in the MACRF parameter of the ACB (option DFR) when shared resources are to be used. The default for both VSAM and the batch LSR subsystem is non-deferred write (NDF). With NDF, VSAM immediately writes a CI, for which a direct PUT is issued, to DASD. When DFR is specified as the LSR option, VSAM does not do the write until:

- Specifically requested to do so, or
- The buffer occupied by the CI is needed for another request, or
- The data set is closed.

If DFR is requested with the batch LSR subsystem, the write of the buffer does not take place until the buffer is needed for another request.

The benefit of DFR is that it can avoid I/O, particularly in cases where there are multiple updates to the same CI (multiple PUTs that resolve to the same CI). This provides a further enhancement to performance beyond what can be gained by LSR alone without DFR.

Deferring writes, however, can have implications which may be prohibitive in certain environments. The areas affected are:

1. **Data set sharing:** When multiple users in different address spaces or systems share the same VSAM cluster, special action on the part of these applications is required to ensure read and write integrity. For details, see the chapter on sharing clusters in *MVS/DFP Using Data Sets*. Some users attempt to share VSAM clusters without taking special application action and may find that this appears to work most of the time. For correct sharing, the updating application must put the updated copy back on DASD, and the reading application must go back to DASD (not a buffer) to get the data subsequently. Failure of either application to take these actions can result in back-level data, or even a “record not found” condition.

   When DFR is used, the exposure to this type of problem is increased. The exposure existed before; this is not a **new** exposure. The possibility of the problem occurring is greater because an updated buffer may stay in storage for a longer period of time before being written.

2. **Recovery considerations:** Some applications assume, for recovery purposes, that when the application regains control after an update request, the update is on DASD (committed). When DFR is used, this assumption is not valid.

For an ordinary batch job, with no other users having access to the cluster at the time the job is run, and with no special recovery requirements (perhaps recovery involves restoring files and rerunning the job), use of DFR should be transparent to the application, except for the potential improved performance.

Dynamic Allocation

Dynamic allocation can be used with the batch LSR subsystem using the standard DALSSNM and DALSSPRM dsname allocation text units. These text units are described in the *MVS/ESA SP V5 Auth Assembler Services Guide*.

The following information reason codes field for error reason code 03A0 pertain to the use of dynamic allocation with batch LSR. The information reason code is returned in the S99INFO field of S99RB, the input parameter request block for dynamic allocation.

See the section on dynamic allocation in the *MVS/ESA SP V5 Auth Assembler Services Guide* for more information.

Figure 3-2. Error Code 03A0 Information Reason Codes and their Meanings

Reason Code	**Meaning**
0001	Undefined parameter
xx02	')' was expected but 'x' was found
xx03	'(' or '=' was expected but 'x' was found
xx04	Value exceeds *vvvvvv*
xx05	Value is less than *vvvvvv*
xx06	Value is not numeric
xx07	Value must be specified
xx08	Value must be specified
xx09	First character is numeric
xx0A	Specified more than once
xx0B	Required but not specified
xx0C	All characters must be alphanumeric or national
000D	Name has more than eight characters
xx0E	Value has more than eight characters
xx0F	Value is the same as the subsystem DDNAME
xx10	Value must be 'E', 'W', or 'I'
xx11	Value must be 'ALL', 'BUFF', 'CB', or 'NONE'

Note: xx is the number of the parameter in error. Use Figure 3-3 on page 3-10 to determine the parameter name associated with the specified parameter number.

Figure 3-3. Table of Parameter Numbers and their Names

Parameter Number	Parameter Name
01	DDNAME
02	BUFND
03	BUFNI
04	HBUFND
05	HBUFNI
06	BUFSD
07	BUFSI
08	SHRPOOL
09	DEFERW
0A	MSG
0B	STRNO
0C	RMODE31

Batch LSR Restrictions and Limitations

There are restrictions and limitations for using the batch LSR subsystem. The LSR buffer management technique is designed for random access, not for sequential processing. For information about improving the performance of jobs that concurrently and sequentially access the same data, see *MVS Hiperbatch Guide*. If you are relying on sequential read-ahead for good performance, using the batch LSR subsystem could degrade, rather than improve, performance. You can issue sequential requests in an LSR environment, but your sequential request will not read in more than one data CI per I/O operation. For NSR, VSAM can read up to a control area's worth of data CIs in a single I/O, conditions permitting.

Note: If you have mixed processing (some direct, some sequential), you may still benefit from using batch LSR. Also, if the amount of data to be processed sequentially is not very large, you can compensate for the lack of read-ahead by using a large data CI size.

The batch LSR subsystem is designed to allow VSAM NSR requests to operate as LSR, without requiring changes to the user program. In general, use of the batch LSR subsystem should be transparent to the application program. However, there are some cases where the batch LSR subsystem cannot be used, and some cases where it can be used but it would be undesirable to do so for performance reasons.

The batch LSR subsystem has the following restrictions:

- The ACB cannot be above 16 megabytes. Otherwise, the system fails the OPEN request with error message IEC190I.
- If the application closes and then reopens the ACB without refreshing the DDNAME, the request is bypassed, and the data set is opened using the same options as the last time it was opened.

  That is, if the data set was previously opened for LSR processing, then it is reopened for LSR. Similarly, if the data set was not eligible for LSR processing the first time, then it is reopened for NSR processing, even if LSR is now applicable.

Note that:

  - High-level languages refresh the DDNAME for each open. Therefore, the subsystem is always called.
  - Even if the subsystem is not called the second time, the application usually wants the same processing (LSR or NSR) method.

- Due to differences in the handling of exclusive control conditions for LSR versus NSR, a program using batch LSR could receive an exclusive control error return code (RPLFDBK=X'14') when it did not receive an error running without batch LSR. This condition could occur only if the application has multiple concurrent 'positions' (RPLs) in the VSAM data set, or is processing a base cluster in its own right while also processing through a path. For details on NSR-LSR differences in handling exclusive control conflicts, see the information about sharing a VSAM data set in *MVS/DFP Using Data Sets*.

Cases Where Batch LSR Cannot Be Used (Detectable by Subsystem)

Certain VSAM options are incompatible with the batch LSR subsystem. If the subsystem detects one of these conditions, it issues a message and bypasses use of the batch LSR subsystem:

Reset option
This option is used with reusable data sets and is specified in the ACB when the user wants reuse to take place. It is indicated on the RST subparameter of the MACRF parameter on the ACB.

User buffering
This option leaves management of I/O buffers up to the user and is specified on the UBF subparameter of the MACRF parameter on the ACB.

Improved control interval processing (ICIP)
This option is specified on the ICI subparameter of the MACRF parameter on the ACB.

Global shared resources (GSR)
Global shared resources (GSR) is specified on the GSR subparameter of the MACRF parameter on the ACB. If the program is already using GSR, the user would not want to use LSR.

Create mode
You cannot open an empty VSAM data set for processing with LSR. However, the subsystem does not have access to the information necessary to determine that the data set is empty without actually opening the data set. Therefore, the user receives an error message if the subsystem attempts to open an empty data set. However, the subsystem then detects the condition, clears the LSR indicators, and reopens the data set for NSR processing.

SHAREOPTIONS 4 with DEFERW=YES
If you specify SHAREOPTIONS 4 with DEFERW=YES when you create the data set, the data set will be processed using batch LSR but not using DFR. If you have specified MSG=I as a SUBSYS subparameter, you will also receive message CSR008I stating that deferred write is not supported with SHAREOPTIONS 4. See *MVS/ESA SP V5 System Messages, Vol 2 (ASB-EWX)* for more information about this message.

Cases Where Batch LSR Cannot Be Used (Not Detectable by Subsystem)

There is another option which is incompatible with batch LSR that the subsystem does not check for because it is not indicated in the ACB:

Implied sequential positioning

With NSR, opening a VSAM file for sequential input provides positioning to the beginning of the file. This does not occur for LSR. If the subsystem is used to convert an application which relies on such positioning to use LSR, the first GET will receive an error indicating that an attempt was made to issue a request which requires positioning without having established a position. This is not detected by the subsystem because it is not detected until the actual GET request is issued.

Cases Where Batch LSR Should Not Be Used For Performance Reasons

Batch LSR may degrade performance when:

- Much of the program's processing is sequential and can benefit from using read-ahead buffers. LSR performs better when the processing is mostly random not sequential. For sequential processing, NSR is the better choice.
- The program rarely accesses a CI more than once. LSR works better when the same CI is repeatedly accessed and the user can provide enough buffers to keep frequently-accessed CIs in central storage across requests. For programs that rarely access the same CI, NSR is the better choice.

S16E Abends While Attempting To Use Batch LSR

There are certain situations where attempts to use batch LSR can cause S16E-14 abends. These situations arise with batch LSR when you are using a DCB instead of an ACB. Three types of situations are identified:

1. COBOL/VS programs using the ISAM interface to access VSAM files.

   There are old COBOL programs which were originally written as ISAM programs. The files may have been converted to VSAM, but the ISAM code is still in the COBOL program. These programs work without batch LSR because VSAM contains a facility called the ISAM interface, which automatically detects that the user is attempting to OPEN an ISAM DCB to a VSAM file, and ‘maps’ the program's ISAM requests to VSAM requests. Using the ISAM interface is incompatible with batch LSR. These programs will receive S16E-14 abends if you use batch LSR for any files which are still coded in the COBOL/VS program as ISAM.

   You may not be aware that a program was originally ISAM. To recognize this situation, inspect the source code of the COBOL/VS program. If the ASSIGN TO clause on the SELECT statement specifies ‘I-ddname’, this is an ISAM program. The COBOL/VS manuals refer to the ‘I’ as an ‘organization’ indicator. For VSAM files, this is omitted.

   Note: This situation is a problem only for COBOL/VS, not for COBOL II. COBOL II does not support ISAM, and ASSIGN TO clauses with the ‘I-ddname’ format will cause compile errors if the COBOL II compiler is used.

To use batch LSR for programs that contain some ISAM coding, you convert the program to specify VSAM. You must convert the program if you will eventually code in COBOL II. In addition to removing the 'I' notation from the ASSIGN TO clause, look for the following COBOL language elements which indicate ISAM:

- APPLY CORE-INDEX
- APPLY REORG-CRITERIA
- File declarations for ISAM
- NOMINAL KEY clause
- TRACK-AREA clause
- USING KEY clause of the START statement.

To help you convert a program, you can use the program offering 'COBOL and CICS Conversion Aid' (5785-ABJ). This program creates COBOL II programs from COBOL/VS programs.

2. PL/1 programs written for ISAM accessing VSAM files.

   Unlike COBOL, PL/1 never uses the ISAM interface. Instead, if the ENVIRONMENT option specifies INDEXED, PL/1 determines whether the file is ISAM or VSAM. If it is VSAM, then VSAM is used without going through the ISAM interface. When batch LSR is requested via the SUBSYS parameter on the JCL, PL/1 cannot determine whether the file is VSAM, resulting in a S16E-14 abend. To correct this situation, change the ENVIRONMENT option from INDEXED to VSAM. This should allow batch LSR to be used.

3. Other programs using the RDJFCB macro to try to identify the file type (for example, IDCAMS)

   No change can be made to circumvent or correct the problem. Batch LSR cannot be used in this case, other than to change the program.

Product-sensitive programming interface

In systems that include MVS/SP 4.3 with PTF UY91649, the JFCBLSR bit in JFCB identifies batch LSR data sets. You can change a program (that uses the RDJFCB macro and checks the JFCORGAM bit to identify VSAM data sets) to treat a data set with a JFCB that has either the JFCORGAM bit or the JFCBLSR bit turned on, as a VSAM data set.

Otherwise, if you are using a different release of MVS or have not applied the PTF, you cannot use batch LSR in this case.

End of Product-sensitive programming interface

Assessing and Tuning With Batch LSR

Once batch LSR candidates are identified, you can assess the value of batch LSR for these selected applications. First, run an application in the usual way without batch LSR. Next, run the application with the default batch LSR parameters. Both of these jobs should include in the jobstream an IDCAMS command, 'LISTCAT ENTRIES (*clustername*) ALL' both before and after the steps you are testing in your application. After running both jobs, compare the EXCP counts in the cluster components before and after the steps you tested. The differences are the number of actual I/O operations to each component that occurred during the step. Next, compare the differences for the original job to the differences when batch LSR is used. If the batch LSR differences are lower, then using batch LSR has eliminated some I/O operations.

For example, these are the LISTCAT EXCP results for a KSDS:

```
   WITHOUT BATCH LSR:
Data component: before step = 12,000 / after step = 20,000
Index component: before step = 15,000 / after step = 24,000

   WITH BATCH LSR:
Data component: before step = 20,000 / after step = 22,000
Index component: before step = 24,000 / after step = 25,000
```

Thus the EXCPs for the non-batch LSR case are:

```
Data component: 8,000
Index component: 9,000
```

The EXCPs for the batch LSR case are:

```
Data component: 2,000
Index component: 1,000
```

The examples show that using batch LSR avoids 6,000 data EXCPs and 8,000 index EXCPs. **Note that these are fictitious numbers which are intended to demonstrate the procedure. They should not be viewed as normal or recommended numbers.**

The same steps are used for tuning testing. Change the batch LSR parameters, rerun the jobs, and compare LISTCAT ALL output with the prior results.

When using batch LSR with hiperspace buffers for a job involving many writes, specify a large number of VSAM (address space) buffers compared to hiperspace buffers. The most frequently referenced data is maintained in address space buffers, while the less frequently referenced data is moved to a hiperspace. A small number of address space buffers will cause increased I/O operations because each time an address space buffer is copied to a hiperspace buffer it is also written to DASD. The fewer the number of address space buffers, the more often the contents must be written to a hiperspace in order to reuse the address space buffer.

Index

Communicating Your Comments to IBM

MVS
Programming:
Batch Local Shared Resources Subsystem Guide

Publication No. GC28-1469-00

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

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.

If you are mailing an RCF from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:
  - FAX: (International Access Code)+1+914+432-9405
- If you prefer to send comments electronically, use this network ID:
  - IBMLink: (United States customers only): KGNVMC(MHVRCFS)
  - IBM Mail Exchange: USIB6TC9 at IBMMAIL
  - Internet e-mail: mhvrcfs@us.ibm.com
  - World Wide Web: http://www.s390.ibm.com/os390

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies

Optionally, if you include your telephone number, we will be able to respond to your comments by phone.

Reader's Comments — We'd Like to Hear from You

MVS
Programming:
Batch Local Shared Resources Subsystem Guide

Publication No. GC28-1469-00

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

Today's date: _____

What is your occupation?

Newsletter number of latest Technical Newsletter (if any) concerning this publication:

How did you use this publication?

[ ]	As an introduction	[ ]	As a text (student)
[ ]	As a reference manual	[ ]	As a text (instructor)
[ ]	For another purpose (explain)		

_____

_____

Is there anything you especially like or dislike about the organization, presentation, or writing in this manual? Helpful comments include general usefulness of the book; possible additions, deletions, and clarifications; specific errors and omissions.

Page Number: Comment:

_____ _____
Name Address

_____ _____
Company or Organization

_____ _____
Phone No.

Reader's Comments — We'd Like to Hear from You
GC28-1469-00

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
522 South Road
Poughkeepsie NY 12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

File Number: S370/S390-34
Program Number: 5655-068
5655-069

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.