CICS/ESA
Performance Guide
Version 4 Release 1

Document Number SC33-1183-02

Note!

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

First edition (October 1994)

This edition applies to Version 4 Release 1 of the IBM licensed program Customer Information Control System/Enterprise Systems Architecture (CICS/ESA), program number 5655-018, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Consult the latest edition of the applicable IBM system bibliography for current information on this product.

This book is based on the Performance Guide for CICS/ESA 3.3, SC33-1183-01. Changes from that edition are marked by vertical lines to the left of the changes.

The CICS/ESA 3.3 edition remains applicable and current for users of CICS/ESA 3.3.

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

At the back of this publication is a page entitled “Sending your comments to IBM”. If you want to make comments, but the methods described are not available to you, please address them to:

IBM United Kingdom Laboratories Limited, Information Development,
Mail Point 095, Hursley Park, Winchester, Hampshire, England, SO21 2JN.

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

Notices

The following paragraph does not apply in any 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 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.

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 that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

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 the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, U.S.A..

Programming Interface Information

This book is intended to help you to:

- Establish performance objectives and monitor them
- Identify performance constraints, and make adjustments to the operational CICS system and its application programs.

This book also documents Product-sensitive Programming Interface and Associated Guidance Information and Diagnosis, Modification or Tuning Information provided by CICS.

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of CICS. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive programming 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 and Associated Guidance Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

Product-sensitive programming interface

Product-sensitive Programming Interface and Associated Guidance Information...

End of Product-sensitive programming interface

Diagnosis, Modification or Tuning Information is provided to help you tune your CICS system.

Warning: Do not use this Diagnosis, Modification or Tuning Information as a programming interface.

Diagnosis, Modification or Tuning Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

Diagnosis, Modification or Tuning Information

Diagnosis, Modification or Tuning Information ...

End of Diagnosis, Modification or Tuning Information

Trademarks and service marks

The following terms, used in this publication, are trademarks or service marks of IBM Corporation in the United States or other countries:

ACF/VTAM
CICS
CICS/ESA
CICS/MVS,
CICSPlex SM
DATABASE 2
DB2
GDDM
Hiperspace
IBM
IMS/ESA
MVS/DFP
MVS/ESA
NetView
System/370
VTAM

Preface

What this book is about

This book is intended to help you to:

- Establish performance objectives and monitor them
- Identify performance constraints, and make adjustments to the operational CICS system and its application programs.

If you need to know where programming interface information is described, or about the definitions of the different types of information in the CICS library, you should read the *CICS Family: Library Guide*.

This book does not discuss the performance aspects of the CICS/ESA Front End Programming Interface. See the *CICS/ESA Front End Programming Interface User's Guide* for more information. This book does not contain Front End Programming Interface dump statistics.

Who this book is for

This book is for a person who is involved in:

- System design
- Monitoring and tuning CICS performance.

What you need to know to understand this book

You need to have a good understanding of how CICS works. This assumes familiarity with most, if not all, of the books in the CICS/ESA library, together with adequate practical experience of installing and maintaining a CICS system.

How to use this book

If you want to establish performance objectives, monitor the performance of a CICS system, and occasionally make adjustments to the system to keep it within objectives, you should read through this book in its entirety.

If you have a performance problem and want to correct it, read Parts 3 and 4 of this book. You may need to refer to various sections in Part 2.

Notes on terminology

The following abbreviations are used throughout this book:

- CICS refers to CICS/ESA.
- VTAM refers to ACF/VTAM.
- DL/I refers to the database component of IMS/ESA.

Bibliography

CICS/ESA 4.1 library

Evaluation and planning	
Release Guide	GC33-1161
Migration Guide	GC33-1162
General	
CICS Family: Library Guide	GC33-1226
Master Index	SC33-1187
User's Handbook	SX33-1188
Glossary (softcopy only)	GC33-1189
Administration	
Installation Guide	SC33-1163
System Definition Guide	SC33-1164
Customization Guide	SC33-1165
Resource Definition Guide	SC33-1166
Operations and Utilities Guide	SC33-1167
CICS-Supplied Transactions	SC33-1168
Program Directory	GC33-1200
Programming	
Application Programming Guide	SC33-1169
Application Programming Reference	SC33-1170
System Programming Reference	SC33-1171
Sample Applications Guide	SC33-1173
Distributed Transaction Programming Guide	SC33-1174
Front End Programming Interface User's Guide	SC33-1175
Diagnosis	
Problem Determination Guide	SC33-1176
Messages and Codes	SC33-1177
Diagnosis Handbook	LX33-6093
Diagnosis Reference	LY33-6082
Data Areas	LY33-6083
Supplementary Data Areas	LY33-6081
Communication	
Intercommunication Guide	SC33-1181
CICS Family: Inter-product Communication	SC33-0824
CICS Family: Communicating from CICS/ESA and CICS/VSE	SC33-0825
Special topics	
Recovery and Restart Guide	SC33-1182
Performance Guide	SC33-1183
CICS-IMS Database Control Guide	SC33-1184
CICS-RACF Security Guide	SC33-1185
Shared Data Tables Guide	SC33-1186
External CICS Interface	SC33-1390

Other CICS books

- *CICS Application Programming Primer (VS COBOL II)*, SC33-0674
- *CICS Application Migration Aid Guide*, SC33-0768
- *Transaction Processing: Concepts and Products*, GC33-0754
- *CICS Family: API Structure*, SC33-1007
- *CICS/ESA XRF Guide* for CICS/ESA Version 3 Release 3, SC33-0661
- *CICS Family: General Information*, GC33-0155
- *CICS/ESA Facilities and Planning Guide* for CICS/ESA Version 3 Release 3, SC33-0654
- *IBM CICS Transaction Affinities Utility MVS/ESA*, SC33-1159

Books from related libraries

ACF/VTAM

ACF/VTAM Version 2 Planning and Installation Reference, SC27-0610
ACF/VTAM Diagnostic Techniques, SY38-3029
IBM CICSPlex System Manager for MVS/ESA Setup and Administration - Volume 2, SC33-0784-01.
IBM 3704 and 3705 Control Program Generation and Utilities Guide, GC30-3008.

DATABASE 2

DATABASE 2 Version 2 Administration Guide, SC26-4374.
DATABASE 2 Performance Monitor (DB2PM) General Information, GH20-6856

IMS/ESA

IMS/ESA Version 3 System Administration Guide, SC26-4282
IMS/ESA Version 3 Database Administration Guide, SC26-4281.

MVS/ESA

MVS/ESA Initialization and Tuning Guide, GC28-1634
MVS/ESA Initialization and Tuning Reference, GC28-1635
MVS/ESA JCL Reference, GC28-1654
MVS/ESA System Modifications, GC28-1636
MVS/ESA System Programming Library: System Management Facilities (SMF), GC28-1819
MVS/ESA System Management Facilities (SMF), GC28-1628
MVS/ESA Installation Exits, GC28-1637

EPDM

Enterprise Performance Data Manager: General Information, GH19-6185
Enterprise Performance Data Manager: Administration Guide, SH19-6816
Enterprise Performance Data Manager/MVS: CICS Performance Feature Guide, SH19-6820

Tuning tools

Generalized Trace Facility Performance Analysis (GTFPARS) Program Description/Operations Manual, SB21-2143
Network Performance Analysis and Reporting System Program Description/Operations, SB21-2488
NetView Performance Monitor (NPM) At A Glance, GH20-6359
Network Program Products Planning, SC30-3351
MVS/ESA RMF General Information, GC28-1028
MVS/ESA RMF User's Guide, GC28-1058
MVS/ESA RMF Version 4 Program Summary, SC28-1023.

Others

IMSASAP II Description/Operations, SB21-1793
Screen Definition Facility II Primer for CICS/BMS Programs, SH19-6118
Systems Network Architecture Management Services Reference,SC30-3346
Teleprocessing Network Simulator General Information, GH20-2487
VTAMPARS II Description/Operations, SB21-2787.

Summary of changes

Changes since CICS/ESA 3.3 are indicated by vertical lines to the left of the text.

Changes for this edition

Changes for the CICS/ESA 4.1 edition include the following:

- Additional or changed statistics in the following areas have been documented:
  - Autoinstalled statistics
  - DBCTL statistics
  - Dispatcher statistics
  - DL/I statistics
  - FEPI pool statistics
  - FEPI connection statistics
  - FEPI target statistics
  - File control statistics
  - ISC/IRC system and mode entry statistics
  - Journal control statistics
  - Loader statistics
  - LSR pool statistics
  - Program autoinstalled statistics
  - Storage manager statistics
  - Suspending mirrors and MROLM
  - Terminal control statistics
  - Terminal autoinstalled statistics
  - Transaction statistics
  - Transaction class statistics
  - Transaction manager statistics
  - Transient data statistics
  - VTAM statistics.
- The domain manager statistics have been removed from this release.
- The description of the data produced by the CICS monitoring facility has been transferred from the *Customization Guide* and is included in “Interpreting CICS monitoring” on page 73.
- Chapter 8, “Enterprise Performance Data Manager/MVS” on page 101 discusses an additional reporting system which uses DB2.
- Chapter 10, “Understanding RMF workload manager data” on page 125 has been added to explain CICS-related data in an RMF workload activity report.
- “Use of LLA (MVS/ESA library lookaside)” on page 189 includes a section on persistent sessions delay interval (PSINT).
- “Intersystems Session Queue Management” on page 281 has been added to “Size of control intervals” on page 217.
- A new appendix has been added giving details of the sample statistics program (DFH0STAT). See Appendix B, “The sample statistics program, DFH0STAT” on page 441.
- The storage chapter has been modified, and new sections about short-on-storage conditions caused by subpool storage fragmentation, and

| kernel storage have been added to Appendix C, “MVS and CICS virtual
| storage” on page 493.

Part 1. Setting performance objectives

This book describes how CICS performance might be improved. It also provides reference information to help you achieve such improvement.

Good performance is the achievement of agreed service levels. This means that system availability and response times meet user's expectations using resources available within the budget.

The performance of a CICS system should be considered:

- When you plan to install a new system
- When you want to review an existing system
- When you contemplate major changes to a system.

There are several basic steps in tuning a system, some of which may be just iterative until performance is acceptable. These are:

1. Agree what good performance is.
2. Set up performance objectives (described in Chapter 1, "Establishing performance objectives").
3. Decide on measurement criteria (described in Chapter 3, "Performance monitoring and review").
4. Measure the performance of the production system.
5. Adjust the system as necessary.
6. Continue to monitor the performance of the system and anticipate future constraints (see "Monitoring for the future" on page 16).

Parts 1 and 2 of this book describe how to monitor and assess performance.

Parts 3 and 4 suggest ways to improve performance.

This part contains the following chapters:

Recommendations given in this book, based on current knowledge of CICS, are general in nature, and cannot be guaranteed to improve the performance of any particular system.

Chapter 1. Establishing performance objectives

Performance objectives often consist of a list of transactions and expected timings for each. Ideally, through them, good performance can be easily recognized and you know when to stop further tuning. They must, therefore, be:

- Practically measurable
- Based on a realistic workload
- Within the budget.

Such objectives may be defined in terms such as:

- Desired or acceptable response times, for example, within which 90% of all responses occur
- Average or peak number of transactions through the system
- System availability, including mean time to failure, and downtime after a failure.

After you have defined the workload and estimated the resources required, you must reconcile the desired response with what you consider attainable. These objectives must then be agreed and regularly reviewed with users.

Establishing performance objectives is an iterative process involving the activities described in the rest of this chapter.

Defining some terms

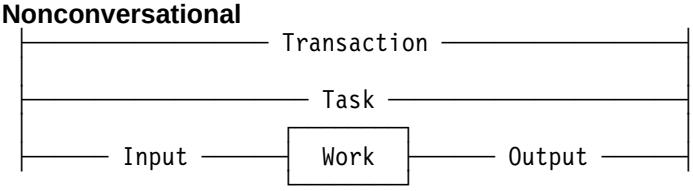
For performance measurements we need to be very specific about what we are measuring. Therefore, it is necessary to define a few terms.

The word **user** here means the terminal operator. A user, so defined, sees CICS performance as the **response time**, that is, the time between the last input action (for example, a keystroke) and the expected response (for example, a message on the screen). Several such responses might be required to complete a user **function**, and the amount of work that a user perceives as a function can vary enormously. So, the number of functions per period of time is not a good measure of performance, unless, of course, there exists an agreed set of benchmark functions.

A more specific unit of measure is therefore needed. The words **transaction** and **task** are used to describe units of work within CICS. Even these can lead to ambiguities, because it would be possible to define transactions and tasks of varying size. However, within a particular system, a series of transactions can be well defined and understood so that it becomes possible to talk about relative performance in terms of transactions per second (or minute, or hour).

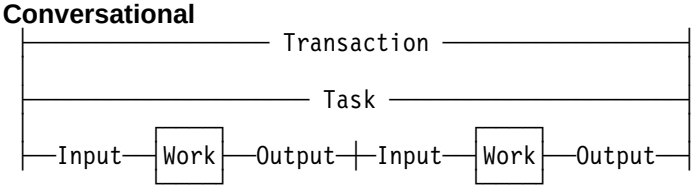
In this context there are three modes of CICS operation.

Nonconversational



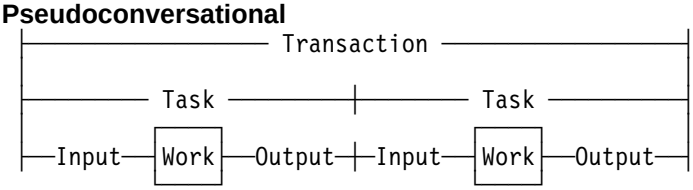
Nonconversational mode is of the nature of one question, one answer; resources are allocated, used, and released immediately on completion of the task. In this mode the words transaction and task are more or less synonymous.

Conversational



Conversational mode is potentially wasteful in a system that does not have abundant resources. There are further questions and answers during which resources are not released. Resources are, therefore, tied up unnecessarily waiting for users to respond, and performance may suffer accordingly. Transaction and task are, once again, more or less synonymous.

Pseudoconversational



Pseudoconversational mode allows for slow response from the user. Transactions are broken up into more than one task, yet the user need not know this. The resources in demand are released at the end of each task, giving a potential for improved performance.

The input/output surrounding a task may be known as the **dialog**.

Defining performance objectives and priorities

Performance objectives and priorities depend on user's expectations. From the point of view of CICS, these objectives state response times to be seen by the terminal user, and the total throughput per day, hour, or minute.

The first step in defining performance objectives is to specify what is required of the system. In doing this, you must consider the available hardware and software resources so that reasonable performance objectives can be agreed. Alternatively you should ascertain what additional resource is necessary to attain users' expectations, and what that resource would cost. This cost might be important in negotiations with users to reach an acceptable compromise between response time and required resource.

An agreement on acceptable performance criteria between the data processing and user groups in an organization is often formalized and called a **service level agreement**.

Common examples in these agreements are, on a network with remote terminals, that 90% of all response times sampled are under six seconds in the prime shift, or that the average response time does not exceed 12 seconds even during peak periods. (These response times could be substantially lower in a network consisting only of local terminals.)

You should consider whether to define your criteria in terms of the average, the 90th percentile, or even the worst-case response time. Your choice may depend on the audit controls of your installation and the nature of the transactions in question.

Analyzing the current workload

Break down the work to be done into transactions Develop a profile for each transaction that includes:

- The **workload**, that is, the amount of work done by CICS to complete this transaction. In an ideal CICS system (with optimum resources), most transactions perform a single function with an identifiable workload.
- The **volume**, that is, the number of times this transaction is expected to be executed during a given period. For an active system, you can get this from the CICS statistics.

Later, transactions with common profiles can be merged, for convenience into **transaction categories**.

Establish the priority of each transaction category, and note the periods during which the priorities change.

Determine the resources required to do the work, that is:

- Physical resources managed by the operating system (real storage, DASD I/O, terminal I/O)
- Logical resources managed by the subsystem, such as control blocks and buffers.

To determine transaction resource demands, you can make sample measurements on a dedicated machine using the CICS monitoring facility. Use these results to suggest possible changes that could have the greatest effect if applied before system-wide contention arises. You can also compare your test results with those in the production environment.

See Chapter 2, “Gathering data for performance objectives” on page 7 for more detailed recommendations on this step.

Translating resource requirements into system objectives

You have to translate the information you have gathered into system-oriented objectives for each transaction category. Such objectives include statements about the transaction volumes to be supported (including any peak periods) and the response times to be achieved.

Any assumptions that you make about your installation must be used consistently in future monitoring. These assumptions include **computing-system factors** and **business factors**.

Computing-system factors include the following:

- **System response time**: this depends on the design and implementation of the code, and the power of the processor.
- **Network response time**: this can amount to seconds, while responses in the processor are likely to be in fractions of seconds. This means that a system can never deliver good responses through an overloaded network, however good the processor.
- **DASD response time**: this is generally responsible for most of the internal processing time required for a transaction. You must consider all I/O operations that affect a transaction.
- **Existing workload**: this may affect the performance of new transactions, and vice versa. In planning the capacity of the system, consider the total load on each major resource, not just the load for the new application.

  Response times can vary for a number of reasons, and the targets should, therefore, specify an acceptable degree of tolerance. Allow for transactions that are known to make heavy demands on the processor and database I/O.

  To reconcile expectations with performance, it may be necessary to change the expectations or to vary the mix or volume of transactions.

Business factors are concerned with work fluctuations. Allow for daily peaks (for example, after receipt of mail), weekly peaks (for example, Monday peak after weekend mail), and seasonal peaks as appropriate to the business. Also allow for the peaks of work after planned interruptions, such as preventive maintenance and public holidays.

Chapter 2. Gathering data for performance objectives

During the design, development, and test of a total system, information is gathered about the complexity of processing with particular emphasis on I/O activity. This information is used for establishing performance objectives.

Four phases of installation planning are suggested:

1. Requirements definition phase
2. External design phase
3. Internal design phase
4. Coding/testing phase.

Requirements definition phase

In this phase, careful estimates are your only input, as follows:

- Number of transactions for each user function
- Number of I/O operations per user function (DASD and terminals)
- Time required to key in user data (including user “thinking time”)
- Line speeds (number of characters per second) for remote terminals
- Number of terminals and operators required to achieve the required rate of input
- Maximum rate of transactions per minute/hour/day/week
- Average and maximum workloads (that is, processing per transaction)
- Average and maximum volumes (that is, total number of transactions)
- Likely effects of performance objectives on operations and system programming.

External design phase

During the external design phase, you should:

1. Estimate the network, processor, and DASD loading based on the dialog between users and tasks (that is, the input to each transaction, and consequent output).
2. Revise your disk access estimates. After external design, only the logical data accesses are defined (for example, EXEC CICS READ).

Remember that, after the system has been brought into service, no amount of tuning can compensate for poor initial design.

Internal design phase

More detailed information is available to help:

- Refine your estimate of loading against the work required for each transaction dialog. Include screen control characters for field formatting.
- Refine disk access estimates against database design. After internal design, the physical data accesses can be defined at least for the application-oriented accesses.
- Add the accesses for CICS temporary storage (scratchpad) data, CICS log, program library, and CICS transient data to the database disk accesses.
- Consider if additional loads could cause a significant constraint.
- Refine estimates on processor use.
- Estimate the internal response time as the sum of I/O times and processor times plus wait times for resources. Allow for paging. For example, if you assume a paging rate of four pages a second, you must allow for as many page faults (typically 25 to 50 milliseconds each) as occur during the time that your transaction is in the system. A page fault in CICS stops all current tasks for the duration of the page fault.

Coding and testing phase

During the coding and testing phase, you should:

1. Refine the internal design estimates of disk and processing resources.
2. Refine the network loading estimates.
3. Run the monitoring tools and compare results with estimates. See Chapter 4, “An overview of performance-measurement tools” on page 25 for information on the CICS monitoring tools.

Post-development review

Review the performance of the complete system in detail. The main purposes are to:

- Validate performance against objectives
- Identify resources whose use requires regular monitoring
- Feed the observed figures back into future estimates.

To achieve this, you should:

1. Identify discrepancies from the estimated resource use
2. Identify the categories of transactions that have caused these discrepancies
3. Assign priorities to remedial actions
4. Identify resources that are consistently heavily used
5. Provide utilities for graphic representation of these resources
6. Project the loadings against the planned future system growth to ensure that adequate capacity is available
7. Update the design document with the observed performance figures

8. Modify the estimating procedures for future systems.

Information supplied by end users

Comments from users are a necessary part of the data for performance analysis and improvement. Reporting procedures must be established, and their use encouraged.

Log exceptional incidents. These incidents should include system, line, or transaction failure, and response times that are outside specified limits. In addition, you should log incidents that threaten performance (such as deadlocks, deadlock abends, stalls, indications of going short-on-storage (SOS) and maximum number of multiregion operation (MRO) sessions used) as well as situations such as recoveries, including recovery from DL/I deadlock abend and restart, which mean that additional system resources are being used.

The data logged should include the date and time, location, duration, cause (if known), and the action taken to resolve the problem.

Chapter 3. Performance monitoring and review

Once set, as described in Chapter 1, “Establishing performance objectives” on page 3, performance objectives should be monitored using appropriate methods. This chapter describes some monitoring techniques; Chapter 4, “An overview of performance-measurement tools” on page 25 describes how to use them.

Deciding on monitoring activities and techniques

In this book, **monitoring** is specifically used to describe regular checking of the performance of a CICS production system, against objectives, by the collection and interpretation of data. Subsequently, **analysis** describes the techniques used to investigate the reasons for performance deterioration. **Tuning** may be used for any actions that result from this analysis.

Monitoring should be ongoing because it:

- Establishes transaction profiles (that is, workload and volumes) and statistical data for predicting system capacities
- Gives early warning through comparative data to avoid performance problems
- Measures and validates any tuning you may have done in response to an earlier performance problem.

A performance history database (see “Enterprise Performance Data Manager (EPDM)” on page 29 for an example) is a valuable source from which to answer questions on system performance, and to plan further tuning.

Monitoring may be described in terms of strategies, procedures, and tasks.

Strategies may include:

- Continuous or periodic summaries of the workload. You can track all transactions or selected representatives.
- Snapshots at normal or peak loads. Peak loads should be monitored for two reasons:
  1. Constraints and slow responses are more pronounced at peak volumes.
  2. The current peak load is a good indicator of the future average load.

Procedures, such as good documentation practices, should provide a management link between monitoring strategies and tasks. The following should be noted:

- The growth of transaction rates and changes in the use of applications
- Consequent extrapolation to show possible future trends
- The effects of nonperformance system problems such as application abends, frequent signon problems, and excessive retries.

Tasks (not to be confused with the task component of a CICS transaction) include:

- Running one or more of the tools described in Chapter 4, “An overview of performance-measurement tools” on page 25
- Collating the output

- Examining it for trends.

You should allocate responsibility for these tasks between operations personnel, programming personnel, and analysts. You must identify the resources that are to be regarded as critical, and set up a procedure to highlight any trends in the use of these resources.

Because the tools require resources, they may disturb the performance of a production system.

Give emphasis to peak periods of activity, for both the new application and the system as a whole. It may be necessary to run the tools more frequently at first to confirm that the expected peaks correspond with the actual ones.

It is not normally practical to keep all the detailed output. Arrange for summarized reports to be filed with the corresponding CICS statistics, and for the output from the tools to be held for an agreed period, with customary safeguards for its protection.

Conclusions on performance should not be based on one or two snapshots of system performance, but rather on data collected at different times over a prolonged period. Emphasis should be placed on peak loading. Because different tools use different measurement criteria, early measurements may give apparently discrepant results.

Your monitoring procedures should be planned ahead of time. These procedures should explain the tools to be used, the analysis techniques to be used, the operational extent of those activities, and how often they are to be performed.

Developing monitoring activities and techniques

When you are developing a master plan for monitoring and performance analysis, you should establish:

- A master schedule of monitoring activity. You should coordinate monitoring with operations procedures to allow for feedback of online events as well as instructions for daily or periodic data gathering.
- The tools to be used for monitoring. The tools used for data gathering should provide for dynamic monitoring, daily collection of statistics, and more detailed monitoring. (See “When to review?” on page 13.)
- The kinds of analysis to be performed. This must take into account any controls you have already established for managing the installation, for example, the use of SLR, and so on. You should document what data is to be extracted from the monitoring output, identifying the source and usage of the data. Although the formatted reports provided by the monitoring tools help to organize the volume of data, you may need to design worksheets to assist in data extraction and reduction.
- A list of the personnel who are to be included in any review of the findings. The results and conclusions from analyzing monitor data should be made known to the user liaison group and to system performance specialists.

- A strategy for implementing changes to the CICS system design resulting from tuning recommendations. This has to be incorporated into installation management procedures, and would include items such as standards for testing and the permitted frequency of changes to the production environment.

Planning the review process

Establish a schedule for monitoring procedures. This schedule should be as simple as possible. The activities done as part of the planning should include the following:

- Listing the CICS requests made by each type of task. This helps you decide which requests or which resources (the high-frequency or high-cost ones) need to be looked at in statistics and CICS monitoring facility reports.
- Drawing up checklists of review questions.
- Estimating resource usage and system loading for new applications. This is to enable you to set an initial basis from which to start comparisons.

When to review?

You should plan for the following broad levels of monitoring activity:

- Dynamic (online) monitoring.
- Daily monitoring.
- Periodic (weekly and monthly) monitoring.
- Keeping sample reports as historical data. You can also keep historical data in the SLR database.

Dynamic monitoring

By dynamic monitoring, we mean “on-the-spot” monitoring that you can, and should, carry out at all times. This type of monitoring generally includes the following:

- Observing the system’s operation continuously to discover any serious short-term deviation from performance objectives.

  Use the CEMT transaction (CEMT INQ|SET MONITOR), together with end-user feedback. You can also use the Resource Measurement Facility (RMF) to collect information about processor, channel, and I/O device usage.
- Obtaining feedback from operators. The control operator is an important source of information about the behavior of the CICS system. An important part of the feedback from the master terminal operator concerns the conditions observed during a CICS monitoring facility run. This information can help validate the data received from the run.
- Obtaining status information. Together with status information obtained by using the CEMT transaction, you can get status information on system processing during online execution. This information could include the queue levels, active regions, active terminals, and the number and type of conversational transactions. You could get this information with the aid of an automated program invoked by the master terminal operator. At prearranged times in the production cycle (such as before scheduling a message, at shutdown of part of the network, or at peak loading), the program could capture the transaction processing status and measurements of system resource levels.

- The System Management product, CICSPlex SM, can accumulate information produced by the CICS monitoring facility to assist in dynamic monitoring activities. The data can then be immediately viewed online, giving instant feedback on the performance of the transactions. To allow CICSPlex SM to collect CICS monitoring information, CICS monitoring must be active using CEMT SET MONITOR ON.

Daily monitoring

The overall objective here is to measure and record key system parameters daily. The daily monitoring data usually consists of counts of events and gross level timings. In some cases, the timings are averaged for the entire CICS system.

- Record both the daily average and the peak period (usually one hour) average of, for example, messages, tasks, processor usage, I/O events, and storage used. Compare these against your major performance objectives and look for adverse trends.
- List the CICS-provided statistics at the end of every CICS run. You should date and time-stamp the data that is provided, and file it for later review. For example, in an installation that has settled down, you might review daily data at the end of the week; generally, you can carry out reviews less frequently than collection, for any one type of monitoring data. If you know there is a problem, you might increase the frequency; for example, reviewing daily data immediately it becomes available.

  You should be familiar with all the facilities in CICS for providing statistics at times other than at shutdown. The main facilities, using the CEMT transaction, are invocation from a terminal (with or without reset of the counters) and automatic time-initiated requests.
- File an informal note of any incidents reported during the run. These may include a shutdown of CICS that causes a gap in the statistics, a complaint from your end users of poor response times, a terminal going out of service, or any other item of significance. This makes it useful when reconciling disparities in detailed performance figures that may be discovered later.
- Print the system console log for the period when CICS was active, and file a copy of the console log in case it becomes necessary to review the CICS system performance in the light of the concurrent batch activity.
- Run one of the performance analysis tools described in Chapter 4, “An overview of performance-measurement tools” on page 25 for at least part of the day if there is any variation in load from day to day. File the summaries of the reports produced by the tools you use.
- Transcribe onto a graph any items identified as being consistently heavily used in the post-development review phase (described in Chapter 2, “Gathering data for performance objectives” on page 7).
- Collect CICS statistics, monitoring data, and RMF data into the SLR database.

Weekly monitoring

Here, the objective is to periodically collect detailed statistics on the operation of your system for comparison with your system-oriented objectives and workload profiles.

- Run the CICS monitoring facility with performance class active, and process it. It may not be necessary to do this every day, but it is important to do it regularly and to keep the sorted summary output as well as the detailed reports.

  Whether you do this on the same day of the week depends on the nature of the system load. If there is an identifiable heavy day of the week, this is the one that you should monitor. (Bear in mind, however, that the use of the monitoring facility causes additional load, particularly with performance class active.)

  If the load is apparently the same each day, run the CICS monitoring facility daily for a period sufficient to confirm this. If there really is little difference from day to day in the CICS load, check the concurrent batch loads in the same way from the logs. This helps you identify any obscure problems because of peak volumes or unusual transaction mixes on specific days of the week. The first few weeks' output from the CICS statistics also give guidance for this.

  It may not be necessary to review the detailed monitor report output every time, but you should always keep this output in case the summary data is insufficient to answer questions raised by the statistics or by user comments. Label the CICS monitoring facility output tape (or a dump of the DASD data set) and keep it for an agreed period in case further investigations are required.
- Run RMF, because this shows I/O usage, channel usage, and so on. File the summary reports and archive the output tapes for some agreed period.
- Review the CICS statistics, and any incident reports.
- Review the graph of critical parameters. If any of the items is approaching a critical level, check the performance analysis and RMF outputs for more detail and follow any previously agreed procedures (for example, notify your management).
- Tabulate or produce a graph of values as a summary for future reference.
- Produce weekly SLR reports.

Monthly monitoring

- Run RMF.
- Review the RMF and performance analysis listings. If there is any indication of excessive resource usage, follow any previously agreed procedures (for example, notify your management), and do further monitoring.
- Date- and time-stamp the RMF output and keep it for use in case performance problems start to arise. You can also use the output in making estimates, when detailed knowledge of component usage may be important. These aids provide detailed data on the usage of resources within the system, including processor usage, use of DASD, and paging rates.
- Produce monthly SLR reports showing long-term trends.

Monitoring for the future

When performance is acceptable, you should establish procedures to monitor system performance measurements and anticipate performance constraints before they become response-time problems. Exception-reporting procedures are a key to an effective monitoring approach.

In a complex production system there is usually too much performance data for it to be comprehensively reviewed every day. Key components of performance degradation can be identified with experience, and those components are the ones to monitor most closely. You should identify trends of usage and other factors (such as batch schedules) to aid in this process.

Consistency of monitoring is also important. Just because performance is good for six months after a system is tuned is no guarantee that it will be good in the seventh month.

Reviewing performance data

The aims of the review procedure are to provide continuous monitoring, and to have a good level of detailed data always available so that there is minimal delay in problem analysis.

Generally, there should be a progressive review of data. You should review daily data weekly, and weekly data monthly, unless any incident report or review raises questions that require an immediate check of the next level of detail. This should be enough to detect out-of-line situations with a minimum of effort.

The review procedure also ensures that additional data is available for problem determination, should it be needed. The weekly review should require approximately one hour, particularly after experience has been gained in the process and after you are able to highlight the items that require special consideration. The monthly review will probably take half a day at first. After the procedure has been in force for a period, it will probably be completed more quickly. However, when new applications are installed or when the transaction volumes or numbers of terminals are increased, the process is likely to take longer.

Review the data from the RMF listings only if there is evidence of a problem from the gross-level data, or if there is an end-user problem that can't be solved by the review process. Thus, the only time that needs to be allocated regularly to the detailed data is the time required to ensure that the measurements were correctly made and reported.

When reviewing performance data, try to:

- Establish the basic pattern in the workload of the installation
- Identify variations from the pattern.

Do not discard ***all*** the data you collect, after a certain period. Discard most, but leave a representative sample. For example, do not throw away ***all*** weekly reports after three months; it is better to save those dealing with the last week of each month. At the end of the year, you can discard all except the last week of each quarter. At the end of the following year, you can discard all the previous year's

data except for the midsummer week. Similarly, you should keep a representative selection of daily figures and monthly figures.

The intention is that you can compare any report for a ***current*** day, week, or month with an ***equivalent*** sample, however far back you want to go. The samples become more widely spaced but do not cease.

Confirming that the system-oriented objectives are reasonable

After the system is initialized and monitoring is operational, you need to find out if the objectives themselves are reasonable (that is, achievable, given the hardware available), based upon actual measurements of the workload.

When you measure performance against objectives and report the results to users, you have to identify any systematic differences between the measured data and what the user sees. This means an investigation of the differences between internal (as seen by CICS) and external (as seen by the end user) measures of response time.

If the measurements differ greatly from the estimates, you must revise application response-time objectives or plan a reduced application workload, or upgrade your system. If the difference is not too large, however, you can embark on tuning the total system. Parts 3 and 4 of this book tell you how to do this tuning activity.

Typical review questions

Use the following questions as a basis for your own checklist. Most of these questions are answered by SLR. See the *SLR User's Guide: Performance Management Version 3*.

Some of the questions are not strictly to do with performance. For instance, if the transaction statistics show a high frequency of transaction abends with usage of the abnormal condition program, this could perhaps indicate signon errors and, therefore, a lack of terminal operator training. This, in itself, is not a performance problem, but is an example of the additional information that can be provided by monitoring.

1. How frequently is each available function used?

   a. Has the usage of transaction identifiers altered?
   b. Does the mix vary from one time of the day to another?
   c. Should statistics be requested more frequently during the day to verify this?

   A different approach must be taken:

   - In systems where all messages are channeled through the same initial task and program (for user security routines, initial editing or formatting, statistical analysis, and so on)
   - For conversational transactions, where a long series of message pairs is reflected by a single transaction
   - In transactions where the amount of work done relies heavily on the input data.

   In these cases, you have to identify the function by program or data set usage, with appropriate reference to the CICS program statistics, file statistics, or other statistics. In addition, you may be able to put user tags into the monitoring data (for example, a user character field in the case of the CICS monitoring facility), which can be used as a basis for analysis by products such as SLR.

   The questions asked above should be directed at the appropriate set of statistics.

2. What is the usage of the telecommunication lines?

   a. Do the CICS terminal statistics indicate any increase in the number of messages on the terminals on each of the lines?
   b. Does the average message length on the CICS performance class monitor reports vary for any transaction type? This can easily happen with an application where the number of lines or fields output depends on the input data.
   c. Is the number of terminal errors acceptable? If you are using a terminal error program or node error program, does this indicate any line problems? If not, this may be a pointer to terminal operator difficulties in using the system.

3. What is the DASD usage?

   a. Is the number of requests to file control increasing? Remember that CICS records the number of logical requests made. The number of physical I/Os depends on the configuration of indexes, and on the data records per control interval and the buffer allocations.
   b. Is intrapartition transient data usage increasing? Transient data involves a number of I/Os depending on the queue mix. You should at least review the number of requests made to see how it compares with previous runs.
   c. Is auxiliary temporary storage usage increasing? Temporary storage uses control interval access, but writes the control interval out only at syncpoint or when the buffer is full.

4. What is the virtual storage usage?
    a. How large are the dynamic storage areas?
    b. Is the number of GETMAIN requests consistent with the number and types of tasks?
    c. Is the short-on-storage (SOS) condition being reached often?
    d. Have any incidents been reported of tasks being purged after deadlock timeout interval (DTIMOUT) expiry?
    e. How much program loading activity is there?
    f. From the monitor report data, is the use of dynamic storage by task type as expected?
    g. Is storage usage similar at each execution of CICS?
    h. Are there any incident reports showing that the first invocation of a function takes a lot longer than subsequent ones? This may arise when programs are loaded that then have to open data sets, particularly in IMS/ESA, for example. Can this be reconciled with application design?
5. What is the processor usage?
    a. Is the processor usage as measured by the monitor report consistent with previous observations?
    b. Are batch jobs that are planned to run, able to run successfully?
    c. Is there any increase in usage of functions running at a higher priority than CICS? Include in this MVS readers and writers, MVS JES, and VTAM if running above CICS, and overall I/O, because of the lower-priority regions.
6. What is the usage of the CICS log?
    a. What is the average output block size written? If it is far below the allocated buffer size, is it possible to reduce the buffer allocation, or does it have to be that large to handle an occasional large item?
    b. What is the I/O rate in requests and physical blocks on the log?
    c. If the log is on tape, is the tape drive fast enough to cope with the traffic required?
    d. If the log is on DASD, is it subject to undue contention from other data sets? In particular, are any of the resources whose updates are logged, resident on the same drive?

       Note: It is bad practice to put a recoverable (updated) resource and a log on the same drive. If that drive fails, you lose both the resource and the log, and you are unable to carry out any forward recovery procedures that you may have.

7. Do any figures indicate design, coding, or operational errors?

    a. Are any of the resources mentioned above heavily used? If so, was this expected at design time? If not, can the heavy use be explained in terms of heavier use of transactions?

    b. Is the heavy usage associated with a particular application? If so, is there evidence of planned growth or peak periods?

    c. Are browse transactions issuing more than the expected number of requests? In other words, is the count of browse requests issued by a transaction greater than what you expected users to cause?

    d. Is the CICS CSAC transaction (provided by the DFHACP abnormal condition program) being used frequently? Is this because invalid transaction identifiers are being entered? For example, errors are signaled if transaction identifiers are entered in lowercase on IBM 3270 terminals but automatic translation of input to uppercase has not been specified.

       A high use of the DFHACP program without a corresponding count of CSAC may indicate that transactions are being entered without proper operator signon. This may, in turn, indicate that some terminal operators need more training in using the system.

In addition to the above, you should regularly review certain items in the CICS statistics, such as:

- Times the MAXTASK limit reached (transaction manager statistics)
- Peak tasks (transaction class statistics)
- Times cushion released (storage manager statistics)
- Storage violations (storage manager statistics)
- Maximum RPLs posted (VTAM statistics)
- Short-on-storage count (storage manager statistics)
- Wait on string total (file control statistics)
- Journal buffers full (journal statistics).

You should also satisfy yourself that large numbers of dumps are not being produced.

Furthermore, you should review the effects of and reasons for system outages and their duration. If there is a series of outages, you may be able to detect a common cause of them.

Anticipating and monitoring system changes and growth

No production system is static. Each system is constantly changing because of new function being added, increased transaction volumes because of a growth in the number of terminal users, addition of new applications or software components, and changes to other aspects of the data processing complex (batch, TSO, and so on). As much as possible, the effects of these changes need to be anticipated, planned for, and monitored.

To find out what application changes are planned, interviewing system or application development managers can be useful in determining the effect of new function or applications and the timing of those changes. Associated with this is the effect of new software to be installed, as well as the known hardware plans for installing new equipment.

When a major change to the system is planned, increase the monitoring frequency before and after the change. A major change includes the addition of:

- A new application or new transactions
- New terminals
- New software releases.

You should look at individual single-thread transactions as well as the overall behavior of the production system.

If the system performance has altered as a result of a major change to the system, data for before-and-after comparison of the appropriate statistics provides the best way of identifying the reasons for the alteration.

Consider having extra tools installed to make it easier to project and test future usage of the system. Tools such as the Teleprocessing Network Simulator (TPNS) program can be used to test new functions under volume conditions before they actually encounter production volumes. Procedures such as these can provide you with insight as to the likely performance of the production system when the changes are implemented, and enable you to plan option changes, equipment changes, scheduling changes, and other methods for stopping a performance problem from arising.

Part 2. Tools that measure the performance of CICS

This part gives an overview of the various tools that can be used to find out which resources are in contention. The first section, Chapter 4, “An overview of performance-measurement tools” on page 25, gives an overview of the tools available. The remaining sections describe the following tools:

- Chapter 5, “Using CICS statistics” on page 41
- Chapter 6, “The CICS monitoring facility” on page 65
- Chapter 7, “Service Level Reporter (SLR)” on page 95
- Chapter 8, “Enterprise Performance Data Manager/MVS” on page 101
- Chapter 9, “MVS Workload Manager” on page 111
- Chapter 10, “Understanding RMF workload manager data” on page 125

Chapter 4. An overview of performance-measurement tools

After reasonable performance objectives have been agreed, you have to set up methods to determine whether the production system is meeting those objectives.

Performance of a production system depends on the usage, paging rates, and virtual storage requirements placed on the main processor, the traffic to and from the disk devices, the traffic of messages throughout the network, and a variety of other factors.

You have to monitor all of these factors to determine when constraints in the system may develop. A variety of programs could be written to monitor all these resources. Many of these programs are currently supplied as part of IBM products such as CICS or IMS/ESA, or are supplied as separate products. This chapter describes some of the products that can give performance information on different components of a production system.

The list of products in this chapter is far from being an exhaustive summary of performance monitoring tools, yet the data provided from these sources comprises a large amount of information. To monitor all this data is an extensive task. Furthermore, only a small subset of the information provided is important for identifying constraints and determining necessary tuning actions, and you have to identify this specific subset for your particular CICS system.

You also have to bear in mind that there are two different types of tools:

1. Tools that directly measure whether you are meeting your objectives
2. Additional tools to look into internal reasons why you might not be meeting objectives.

None of the tools can directly measure whether you are meeting end-user response time objectives. The lifetime of a task within CICS is comparable, that is, usually related to, response time, and bad response time is usually correlated with long lifetime within CICS, but this correlation is not exact because of other contributors to response time.

Obviously, you want tools that help you to measure your objectives. In some cases, you may choose a tool that looks at some internal function that contributes towards your performance objectives, such as task lifetime, rather than directly measuring the actual objective, because of the difficulty of measuring it.

When you have gained experience of the system, you should have a good idea of the particular things that are most significant in that particular system and, therefore, what things might be used as the basis for exception reporting. Then, one way of simply monitoring the important data might be to set up exception-reporting procedures that filter out the data that is not essential to the tuning process. This involves setting standards for performance criteria that identify constraints, so that the exceptions can be distinguished and reported while normal performance data is filtered out. These standards vary according to individual system requirements and service level agreements.

You often have to gather a considerable amount of data before you can fully understand the behavior of your own system and determine where a tuning effort can provide the best overall performance improvement. Familiarity with the analysis tools and the data they provide is basic to any successful tuning effort.

Remember, however, that all monitoring tools cost processing effort to use. Typical costs are 5 to 10% additional processor cycles for the CICS monitoring facility (performance class), and up to 1% for the exception class. The CICS trace facility overhead is highly dependent on the workload used. The overhead can be in excess of 25%.

In general, then, we recommend that you use the following tools in the sequence of priorities shown below:

1. CICS statistics
2. CICS monitoring data
3. CICS internal and auxiliary trace.

In this chapter, the overview of the various tools for gathering or analyzing data is arranged as follows:

- **CICS performance data**
- **Operating system performance data**
- **Performance data for other products**.

CICS performance data

CICS statistics

CICS statistics are the simplest and the most important tool for permanently monitoring a CICS system. They collect information on the CICS system as a whole, without regard to tasks.

The CICS statistics domain writes five types of statistics to SMF data sets: **interval, end-of-day, requested, requested reset**, and **unsolicited** statistics.

Each of these sets of data is described and a more general description of CICS statistics is given in Appendix A, "CICS statistics tables" on page 329.

The CICS monitoring facility

The CICS monitoring facility collects information about CICS tasks, and is described more completely in Chapter 6, "The CICS monitoring facility" on page 65.

The *CICS/ESA Customization Guide* contains programming information on the data set formats and the *CICS/ESA Operations and Utilities Guide* describes the monitoring utility programs, DFHMNDUP and DFH$MOLS.

The sample statistics program (DFH0STAT)

You can use the statistics sample program, DFH0STAT, to help you determine and adjust the values needed for CICS/ESA storage parameters, for example, using DSALIM and EDSALIM. The program produces a report showing critical system parameters from the CICS/ESA dispatcher, an analysis of the CICS/ESA storage manager and loader statistics, and an overview of the MVS storage in use. The program demonstrates the use of the EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of a CICS/ESA system. You can use the sample program as provided or modify it to suit your needs. It can be used to provide data about the following:

- System Status, Monitoring and Statistics
- Transaction Manager and Dispatcher
- Storage
- Loader
- Transactions
- Transaction Totals including Subspace usage information
- Programs
- Program Totals
- DFHRPL Analysis
- Temporary Storage
- Transient Data
- User Domain
- LSR Pools
- Files
- Data Tables.

See Appendix B, “The sample statistics program, DFH0STAT” on page 441 for the details and interpretation of the report.

CICS trace facilities

For the more complex problems that involve system interactions, you can use the CICS trace to record the progress of CICS transactions through the CICS management modules. Whereas a dump gives a “snapshot” of conditions at a particular moment, CICS trace provides a history of events leading up to a specific situation. CICS includes facilities for selective activation or deactivation of some groups of traces.

The CICS trace facilities can also be useful for analyzing performance problems such as excessive waiting on events in the system, or constraints resulting from inefficient system setup or application program design.

Several types of tracing are provided by CICS, and are described in the *CICS/ESA Problem Determination Guide*. Trace is controlled by:

- The system initialization parameters (see the *CICS/ESA System Definition Guide*).
- CETR (see the *CICS/ESA CICS-Supplied Transactions* manual). CETR also provides for trace selectivity by, for instance, transaction type or terminal name.
- CEMT SET INTRACE, CEMT SET AUXTRACE, or CEMT SET GTFTRACE (see the *CICS/ESA CICS-Supplied Transactions* manual).

- EXEC CICS SET TRACEDEST, EXEC CICS SET TRACEFLAG, or EXEC CICS SET TRACETYPE (see the *CICS/ESA System Programming Reference* for programming information).

Three destinations are available for trace data:

1. The internal trace table, in main storage above the 16MB line
2. Auxiliary trace data sets, defined as BSAM data sets on tape or disk
3. The MVS generalized trace facility (GTF) data sets, which can be accessed through the MVS interactive problem control system (IPCS).

Other CICS data

The measurement tools previously described do not provide all the data necessary for a complete evaluation of current system performance. They do not provide information on how and under what conditions each resource is being used, nor do they provide information about the existing system configuration while the data is being collected. It is therefore extremely important to use as many techniques as possible to get information about the system. Additional sources of information include the following:

- Hardware configuration
- VTOC listings
- LISTCAT (VSAM)
- CICS table listings, especially:
  - SIT (and overrides in the CICS startup procedure)
  - FCT (file control table)
  - VSAM file specifications
- Link pack area (LPA) map
- Load module cross-reference of the CICS nucleus
- SYS1.PARMLIB listing
- Dump of the CICS address space. See the *CICS/ESA Operations and Utilities Guide* for information on how to get an address space dump for CICS when the CICS address space abends.

This data, used with the data produced by the measurement tools, provides the basic information that you should have for evaluating your system's performance.

Operating system performance data

- "Enterprise Performance Data Manager (EPDM)"
- "Generalized trace facility (GTF)" on page 30
- "Resource measurement facility (RMF)" on page 32.

Enterprise Performance Data Manager (EPDM)

EPDM is an IBM product that collects and analyzes data from CICS and other IBM systems and products. With EPDM you can build reports which help you with the following:

- System overviews
- Service levels
- Availability
- Performance and tuning
- Capacity planning
- Change and problem management
- Accounting.

A large number of ready-made reports are available, and in addition you can generate your own reports to meet specific needs.

EPDM's reports use data from CICS monitoring and statistics. EPDM also collects data from the MVS system and from products such as RMF, TSO, IMS and NetView. This means that data from CICS and other systems can be shown together, or can be presented in separate reports.

Reports can be presented as plots, bar charts, pie charts, tower charts, histograms, surface charts, and other graphic formats. EPDM simply passes the data and formatting details to Graphic Data Display Manager (GDDM6) which does the rest. EPDM can also produce line graphs and histograms using character graphics where GDDM is not available, or the output device does not support graphics. For some reports, where you need the exact figures, numeric reports such as tables and matrices are more suitable.

See Chapter 8, "Enterprise Performance Data Manager/MVS" on page 101 for more information on EPDM as a CICS performance measurement tool.

Generalized trace facility (GTF)

As described above, CICS trace entries can be recorded via GTF, and reports produced via IPCS. More generally, GTF is an integral part of the MVS system, and traces the following system events: DASD seek addresses on start I/O instructions, system resources manager (SRM) activity, page faults, I/O activity, and supervisor services. Execution options specify the system events to be traced. The amount of processing time to be used by GTF can vary considerably, depending on the number of events to be traced. You should request the time-stamping of GTF records with the TIME=YES operand on the EXEC statement for all GTF tracing.

GTF should run at a dispatching priority (DPRTY) of 255 so that records are not lost. If GTF records are lost and the DPRTY is specified at 255, specify the BUF operand on the execute statement as greater than 10 buffers.

GTF is generally used to monitor short periods of system activity and you should run it accordingly.

You can use these options to get the data normally needed for CICS performance studies:

```
TRACE=SYS,RNIO,USR     (VTAM)
TRACE=SYS              (Non-VTAM)
```

If you need data on the units of work dispatched by the system and on the length of time it takes to execute events such as SVCs, LOADs, and so on, the options are:

```
TRACE=SYS,SRM,DSP,TRC,PCI,USR,RNIO
```

The TRC option produces the GTF trace records that indicate GTF interrupts of other tasks that it is tracing. This set of options uses a higher percentage of processor resources, and you should use it only when you need a detailed analysis or timing of events.

No data-reduction programs are provided with GTF. To extract and summarize the data into a meaningful and manageable form, you can either write a data-reduction program or use one of the program offerings that are available.

For further details, see the *MVS/ESA Initialization and Tuning Guide*.

GTF reports

You can produce reports from GTF data using the interactive problem control system (IPCS). The reports generated by IPCS are useful in evaluating both system and individual job performance. It produces job and system summary reports as well as an abbreviated detail trace report. The summary reports include information on MVS dispatches, SVC usage, contents supervision, I/O counts and timing, seek analysis, page faults, and other events traced by GTF. The detail trace reports can be used to follow a transaction chronologically through the system.

Other reports are available that:

- Map the seek addresses for a specific volume
- Map the arm movement for a specific volume
- Map the references to data sets and members within partitioned data sets
- Map the page faults and module reference in the link pack area (LPA).

These reports are described later in this section.

Before GTF is run, you should plan the events to be traced. If specific events such as start I/Os (SIOs) are not traced, and the SIO-I/O timings are required, the trace must be re-created to get the data needed for the reports.

If there are any alternative paths to a control unit in the system being monitored, you should include the PATHIO input statement in the report execution statement. Without the PATHIO operand, there are multiple I/O lines on the report for the device with an alternative path: one line for the primary device address and one for the secondary device address. If this operand is not included, the I/Os for the primary and alternate device addresses have to be combined manually to get the totals for that device.

Seek histogram report

The seek histogram report (SKHST) can help you find out if there is any arm contention on that volume, that is, if there are any long seeks on the volume being mapped. It produces two reports: the first shows the number of seeks to a particular address, and the second shows the distance the arm moves between seeks. These reports can be used to determine if you should request a volume map report to investigate further the need to reorganize a specific volume.

Volume map report

The volume map report (VOLMAP) displays information about data sets on the volume being mapped and about seek activity to each data set on that volume. It also maps the members of a partitioned data set and the count of seeks issued to each member. This report can be very useful in reorganizing the data sets on a volume and in reorganizing the members within a partitioned data set to reduce the arm movement on that specific volume.

Reference map report

The reference map report (REFMAP) shows the page fault activity in the link pack area (LPA) of MVS. This reference is by module name and separates the data faults from the instruction faults. The report also shows the count of references to the specific module. This reference is selected from the address in the stored PSW of the I/O and EXT interrupt trace events from GTF. This report can be useful if you want to make changes to the current MVS pack list in order to reduce real storage or to reduce the number of page faults that are being encountered in the pageable link pack area of MVS.

Resource measurement facility (RMF)

The Resource Measurement Facility (RMF) collects system-wide data that describes the processor activity (WAIT time), I/O activity (channel and device usage), main storage activity (demand and swap paging statistics), and system resources manager (SRM) activity (workload).

RMF Version 4 is a centralized measurement tool that monitors system activity to collect performance and capacity planning data. The analysis of RMF reports provides the basis for tuning the system to user requirements. They can also be used to track resource usage.

RMF Version 4 measures the following activities:

- Processor usage
- Address space usage
- Channel activity:
  - Request rate and service time per physical channel
  - Logical-to-physical channel relationships
  - Logical channel queue depths and reasons for queuing.
- Device activity and contention for the following devices:
  - Unit record
  - Graphics
  - Direct access storage
  - Communication equipment
  - Magnetic tapes
  - Character readers.
- Detailed system paging
- Detailed system workload
- Page and swap data set
- Enqueue.

RMF Version 4 allows the MVS/ESA user to:

- Evaluate system responsiveness:
  - Identify bottlenecks. The detailed paging report associated with the page and swap data set activity can give a good picture of the behavior of a virtual storage environment.

- Check the effects of tuning:
  - Results can be observed dynamically on a screen or by postprocessing facilities.
- Perform capacity planning evaluation:
  - The workload activity reports include the interval service broken down by key elements such as processor, input/output, and main storage service.
  - Analysis of the resource monitor output (for example, system contention indicators, swap-out broken down by category, average ready users per domain) helps in understanding user environments and forecasting trends.
  - The post-processing capabilities make the analysis of peak load periods and trend analysis easier.
- Manage the larger workloads and increased resources that MVS/ESA can support
- Identify and measure the usage of online channel paths
- Optimize the usefulness of expanded storage capability.

RMF measures and reports system activity and, in most cases, uses a sampling technique to collect data. Reporting can be done with one of three monitors:

1. Monitor I measures and reports the use of system resources (that is, the processor, I/O devices, storage, and data sets on which a job can enqueue during its execution). It runs in the background and measures data over a period of time. Reports can be printed immediately after the end of the measurement interval, or the data can be stored in SMF records and printed later with the RMF postprocessor. The RMF postprocessor can be used to generate reports for “exceptions”: conditions where user-specified values are exceeded.
2. Monitor II, like Monitor I, measures and reports the use of system resources. It runs in the background under TSO or on a console. It provides “snapshot” reports about resource usage, and also allows its data to be stored in SMF records. The RMF postprocessor can be used to generate exception reports.
3. Monitor III primarily measures the contention for system resources and the delay of jobs that such contention causes. It collects and reports the data in real time at a display station, with optional printed copy backup of individual displays. Monitor III can also provide exception reports, but its data ***cannot*** be stored in SMF records.

RMF should be active in the system 24 hours a day, and you should run it at a dispatching priority above other address spaces in the system so that:

- The reports are written at the interval requested
- Other work is not delayed because of locks held by RMF.

A report is generated at the time interval specified by the installation. The largest system overhead of RMF occurs during the report generation: the shorter the interval between reports, the larger the burden on the system. An interval of 60 minutes is recommended for normal operation. When you are addressing a specific problem, reduce the time interval to 10 or 15 minutes. The RMF records can be directed to the SMF data sets with the NOREPORT and RECORD options; the report overhead is not incurred and the SMF records can be formatted later.

Note: There may be some discrepancy between the CICS initialization and termination times when comparing RMF reports against output from the CICS monitoring facility.

For further details of RMF Version 4, see the *MVS/ESA Resource Measurement Facility (RMF) Version 4 Program Summary* and the *MVS/ESA Resource Measurement Facility (RMF) General Information* manual.

Guidance on how to use RMF with the CICS monitoring facility is given in "Using CICS monitoring SYSEVENT information with RMF" on page 67. In terms of CPU costs this is an inexpensive way to collect performance information. Shorter reports throughout the day are needed for RMF because a report of a full day's length includes startup and shutdown and does not identify the peak period.

Performance data for other products

This section gives an overview of the tools that can be used to monitor information on various access methods and other programs used with CICS and the operating system.

ACF/VTAM

ACF/VTAM (program number 5735-RC2) provides information about buffer usage either to GTF in SMF trace data or to the system console through DISPLAY and BFRUSE commands. Other tuning statistics can also be recorded on the system console through the MODIFY procname, TNSTAT command. (This command is described in the *ACF/VTAM Diagnostic Techniques* manual.)

Virtual telecommunication access method (VTAM) trace

The VTAM trace facility is provided as part of VTAM, and tracks messages through different points to and from CICS. The time-stamps that are included can be particularly useful in determining where a transaction spends large amounts of time.

Network performance, analysis, and reporting system (NETPARS)

NETPARS is a program offering (program number 5798-CZX) that analyzes network log data from the NetView Performance Monitor (NPM). Further information on NETPARS is given in the *Network Performance, Analysis, and Reporting System (NETPARS) Description/Operations*.

VTAM performance, analysis, and reporting system II (VTAMPARS II)

The VTAMPARS program offering (program number 5798-DFE) provides information on network traffic through the VTAM component of a network. Information on terminal connect time, message characteristics and rates, and so forth, can be collected and analyzed. Further information on VTAMPARS is given in the *VTAM Performance, Analysis, and Reporting System (VTAMPARS) Program Description/Operations*.

Generalized performance analysis reporting (GPAR)

Generalized Performance Analysis Reporting (GPAR) (program number 5798-CPR) is a prerequisite for VTAMPARS. GPAR is designed as a base for reporting programs (IBM or user-written). It helps summarize sequential activity traces like GTF traces. It also contains facilities to print user-tailored graphs from any performance data log or non-VSAM sequential data set.

VTAM storage management (SMS) trace

The VTAM storage management (SMS) trace facility collects information on VTAM's usage of its buffers, including which buffers are used in the various buffer pools, and the number of buffer expansions and depletions.

VTAM tuning statistics

Information provided in the VTAM tuning statistics includes data on the performance between VTAM and the network control program (NCP), the number of reads and writes and what caused that activity, and message counts.

NetView for MVS/ESA

NetView is a network management program offering (program number 5665-362) which provides a cohesive set of SNA host network management services in a single product. NetView includes the functions of the network communication control facility (NCCF), network logical data manager (NLDM), and network problem determination application (NPDA), as well as functions of the VTAM node control application (VNCA) and network management productivity facility (NMPF). Support is provided for problem determination of the IBM 3720 Communication Controller and for online configuration control and testing of IBM 586X modems.

NetView's set of network management functions consists of the following:

- Command facility
- Session monitor
- Hardware monitor
- Status monitor
- Online HELP and Help Desk facility
- Browse facility.

NetView's capabilities include:

- Terminal access facility support of large screen and color applications (for example, the NetView performance monitor)
- CLISTs driven by application messages
- Disk log enhancements
- Support for 586X Models 2 and 3 and 5812 modems
- Token-ring network support

- Virtual route blockage indication
- Session setup failure notification
- Extended recovery facility
- Automatic operations and recovery
- Real-time update of the domain status panel
- Easy-to-use installation procedure.

The benefits provided by NetView include:

- Improved cohesion and usability in support of network management functions
- Enhanced installation, operation, and utilization of network management functions in MVS/ESA environments.

For further information on NetView, see the *Systems Network Architecture Management Services Reference*, and *Network Program Products Planning*.

NetView performance monitor (NPM)

The NetView Performance Monitor (NPM) program product (program number 5665-333) is designed to aid network support personnel in managing VTAM-based communications networks. It collects and reports on data in the host and NCP.

NPM data can be used to:

- Identify network traffic bottlenecks
- Display screens showing volume and response times for various resources
- Generate color graphs of real-time and historical data
- Alert users to response time threshold overages.

NPM performance data can also help to:

- Determine the performance characteristics of a network and its components
- Identify network performance problems
- Tune communications networks for better performance as well as verify the effects of problem resolutions
- Gauge unused capacity when planning for current network changes
- Produce timely and meaningful reports on network status for multiple levels of management.

Further information on NPM is given in *NetView Performance Monitor At A Glance*.

LISTCAT (VSAM)

VSAM LISTCAT provides information that interprets the actual situation of VSAM data sets. This information includes counts of the following:

- Whether and how often control interval (CI) or control area (CA) splits occur (splits should occur very rarely, especially in CA).
- Physical accesses to the data set.
- Extents for a data set (secondary allocation). You should avoid this secondary allocation, if possible, by making the primary allocation sufficiently large.
- Index levels.

Virtual storage access method (VSAM) or ICF catalog

Information kept in the VSAM or Integrated Catalog Facility (ICF) catalog includes items on record sizes, data set activity, and data set organization.

DB monitor (IMS)

The IMS DB monitor report print program (DFSUTR30) provides information on batch activity (a single-thread environment) to IMS databases, and is activated through the DLMON system initialization parameter. As in the case of CICS auxiliary trace, this is for more in-depth investigation of performance problems by single-thread studies of individual transactions.

The DB monitor cannot be started and stopped from a terminal. After the DB monitor is started in a CICS environment, the only way to stop it is to shut down CICS. The DB monitor cannot be started or stopped dynamically.

When the DB monitor runs out of space on the IMSMON data set, it stops recording. The IMSMON data set is a sequential data set, for which you can allocate space with IEFBR14. The DCB attributes are:

```
DCB=(RECFM=VB,LRECL=2044,BLKSIZE=2048)
```

If you are running the DB monitor in a multithread (more than one) environment, the only statistics that are valid are the VSAM buffer pool statistics.

Program isolation (PI) trace

The program isolation (PI) trace can point out database contention problems arising from the nature of task's access to a particular database. Because only one task can have access to a record at one time, and any other task waits till the record is freed, high contention can mean high response time. This trace is part of IMS, and can be activated by the CEMT SET PITRACE ON|OFF command. Information on the format of the PI trace report is given in the *IMS/ESA Version 3 System Administration Guide*.

IMS System Utilities/Database Tools (DBT)

The IMS System Utilities/Database Tools (DBT) program product (program number 5668-856) is a powerful package of database programs and products designed to enhance data integrity, data availability, and performance of IMS databases. It provides the important tools that are needed to support both full-function (index, HDAM, HIDAM, and HISAM) and fastpath (DEDB) databases.

DBT can help you maintain data integrity by assisting the detection and repair of errors before a problem disrupts operations. It speeds database reorganization by providing a clear picture of how data is stored in the database, by allowing the user to simulate various database designs before creating a new database, and by providing various sort, unload, and reload facilities. DBT also improves programming productivity by providing monitoring capabilities and by reducing the need to write reformatting programs. It increases the user's understanding of the database for analysis, tuning, and reorganization. It also helps enhance the overall database performance.

For further information, see the *IMS System Utilities/Database Tools (DBT) General Information* manual.

IMS monitor summary and system analysis II (IMSASAP II)

IMSASAP II (program number 5798-CHJ) is a performance analysis and tuning aid for IMS/ESA database and data communication systems. It is a report program that executes under IMS/ESA for Generalized Performance Analysis Reporting (GPAR). IMSASAP II processes IMS/ESA DB and DC monitor data to provide summary, system analysis, and program analysis level reports that assist in the analysis of an IMS/ESA system environment. The monitor concept has proven to be a valuable aid in the performance analysis and tuning of IMS systems. IMSASAP II extends this capability by providing comprehensive reports (from management summaries to detail program traces) to meet a broad range of IMS/ESA system analysis objectives.

IMSASAP:

- Produces a comprehensive set of reports, organized by level of detail and area of analysis, to satisfy a wide range of IMS/ESA system analysis requirements
- Provides report selection and reporting options to satisfy individual requirements and to assist in efficient analysis
- Produces alphanumerically collated report items in terms of ratios, rates, and percentages to facilitate a comparison of results without additional computations
- Reports on schedules in progress including wait-for-input and batch message processing programs
- Provides reports on IMS/ESA batch programs.

Further information on IMSASAP is given in the *IMSASAP II Program Description/Operations* manual.

DATABASE 2 Performance Monitor (DB2PM)

DATABASE 2 Performance Monitor (program number 5665-354) analyses DB2 performance data and generates a comprehensive set of reports. These include the following:

- A set of graphs showing DB2 statistics, accounting, and frequency distribution performance data
- A summary of DB2 system activity, including system tasks (statistics data)
- A summary of DB2 application work, reported either by user or by application (accounting data)
- A set of transit time reports detailing DB2 workload performance
- System- and application-related DB2 I/O activity
- Locking activity, reported both by DB2 application type and by database
- SQL activity
- Selective tracing and formatting of DB2 records.

For further information, see the *DATABASE 2 Performance Monitor (DB2PM) General Information* manual.

Teleprocessing network simulator (TPNS)

The Teleprocessing Network Simulator (TPNS) (program number 5662-262) is a program that simulates terminal activity such as that coming through the NCP. TPNS can be used to operate an online system at different transaction rates, and can monitor system performance at those rates. TPNS also keeps information on response times, which can be analyzed after a simulation.

Further information on TPNS is given in the *Teleprocessing Network Simulator (TPNS) General Information* manual.

Chapter 5. Using CICS statistics

This chapter introduces and describes CICS statistics. It describes the methods for collecting statistics and discusses those statistics that can be used for tuning your CICS system.

Introduction to CICS statistics

CICS management modules control how events are managed by CICS. As events occur, CICS produces information that is available to you as system and resource statistics.

The resources controlled by CICS include files, databases, journals, transactions, programs, and tasks. Resources that CICS manage, and values that CICS use in its record-keeping role, are defined in one of the following ways:

- Online, by the CICS CEDA transaction.
- Offline, by the CICS system definition (CSD) utility program, DFHCSDUP. See the *CICS/ESA Customization Guide* for programming information about DFHCSDUP.
- Offline, by CICS control table macros.

Statistics are collected during CICS online processing for later offline analysis. The statistics domain writes statistics records to a System Management Facilities (SMF) data set. The records are of SMF type 110, sub-type 002. Monitoring records and some journaling records are also written to the SMF data set as type 110 records. You might find it useful to process statistics and monitoring records together. For programming information about SMF, and about other SMF data set considerations, see the *CICS/ESA Customization Guide*.

Types of statistics data

CICS produces five types of statistics:

Interval statistics
are gathered by CICS during a specified interval. CICS writes the interval statistics to the SMF data set automatically at the expiry of the interval if:

- Statistics recording status was set ON by the STATRCD system initialization parameter (and has not subsequently been set OFF by a CEMT or EXEC CICS SET STATISTICS RECORDING command). The default is STATRCD=OFF.
- ON is specified in CEMT SET STATISTICS.
- The RECORDING option of the EXEC CICS SET STATISTICS command is set to ON.

End-of-day statistics
are a special case of interval statistics where all statistics counters are collected and reset. There are three ways to reset the statistics counters:

- The end-of-day expiry time

- When CICS quiesces (normal shutdown)
- When CICS terminates (immediate shutdown).

The end of day value defines a logical point in the 24 hour operation of CICS. You can change the end of day value using CEMT SET STATISTICS or the EXEC CICS SET STATISTICS command.
End-of-day statistics are always written to the SMF data set, regardless of the settings of any of the following:

- The system initialization parameter, STATRCD, or
- CEMT SET STATISTICS or
- The RECORDING option of EXEC CICS SET STATISTICS.

The statistics that are written to the SMF data set are those collected since the last event which involved a reset. The following are examples of resets:

- At CICS startup
- Issue of RESETNOW RECORDNOW in CEMT or EXEC CICS STATISTICS commands.

The default end-of-day value is 000000 (midnight).

Requested statistics
are statistics that the user has asked for by using one of the following commands:

- CEMT PERFORM STATISTICS RECORD
- EXEC CICS PERFORM STATISTICS RECORD
- EXEC CICS SET STATISTICS ON|OFF RECORDNOW

These commands cause the statistics to be written to the SMF data set immediately, instead of waiting for the current interval to expire. The PERFORM STATISTICS command can be issued with any combination of resource types or you can ask for all resource types with the ALL option. For more details about CEMT commands see the *CICS/ESA CICS-Supplied Transactions*; for programming information about the equivalent EXEC CICS commands, see the *CICS/ESA System Programming Reference*.

End-of-day statistics are always written to the SMF data set, regardless of the settings of any of the following:

- The system initialization parameter, STATRCD, or
- CEMT SET STATISTICS or
- The RECORDING option of EXEC CICS SET STATISTICS.

Requested reset statistics
differ from requested statistics in that all statistics are collected and statistics counters are reset. There are three ways to reset the statistics counters:

- CEMT PERFORM STATISTICS RECORD ALL RESETNOW
- EXEC CICS PERFORM STATISTICS RECORD ALL RESETNOW
- EXEC CICS SET STATISTICS ON|OFF RESETNOW RECORDNOW

You can also invoke requested reset statistics when changing the recording status from ON to OFF, or vice versa, using CEMT SET

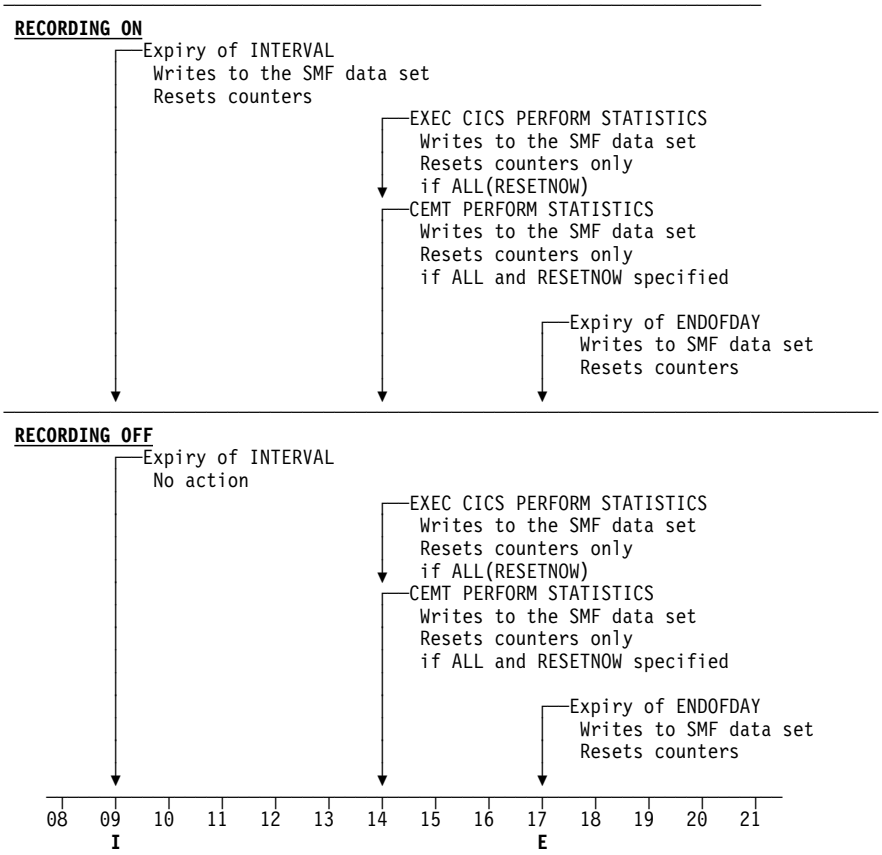


Figure 1. Summary of statistics reset functions

STATISTICS ON|OFF RECORDNOW RESETNOW, or EXEC CICS SET STATISTICS ON|OFF RECORDNOW RESETNOW.

Note: It is valid to specify RECORDNOW RESETNOW options only when there is a genuine change of status from STATISTICS ON to OFF, or vice versa. In other words, coding EXEC CICS SET STATISTICS ON RECORDNOW RESETNOW when statistics is already ON will cause an error response.

The PERFORM STATISTICS command must be issued with the ALL option if RESETNOW is present.

RESETNOW RECORDNOW on the SET STATISTICS command can only be invoked if the RECORDING option is changed. See also Figure 1.

Note: Issuing the RESETNOW command by itself in the SET STATISTICS command causes the loss of the statistics data that has been collected since the last interval. Interval collections take place only if you set the RECORDING status ON. To set the statistics recording status ON or OFF, use either the RECORDING option on this command or the SIT parameter STATRCD. Statistics are always written, and counts reset, at the end of day. See Figure 1 for further information.

Unsolicited statistics
are automatically gathered by CICS for dynamically allocated and deallocated resources. CICS writes these statistics to SMF just before the resource is deleted, regardless of the status of statistics recording.

CICS collects unsolicited statistics for:

Autoinstall
Whenever an autoinstalled terminal entry in the TCT is deleted (after the terminal logs off), CICS collects statistics covering the autoinstalled period since the last interval. The period covers any delay interval specified by the system initialization parameter, AILDELAY.

If an autoinstall terminal logs on again before the expiry of the delay interval, then the accumulation of statistics continues until the next interval. At that interval, the accumulation of statistics is restarted.

DBCTL Whenever CICS disconnects from DBCTL, CICS collects the statistics covering the whole of the DBCTL connection period.

FEPI nodes
Whenever an installed FEPI node definition is discarded, CICS collects the statistics covering the installed period since the last interval.

FEPI pools
Whenever an installed FEPI pool definition is discarded, CICS collects the statistics covering the installed period since the last interval.

FEPI targets
Whenever an installed FEPI target definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Files Whenever CICS closes a file, CICS collects statistics covering the period from the last interval.

Whenever CICS closes a file which is in an LSRPOOL, LSRPOOL file statistics are collected (in addition to the file statistics), covering the period from the last interval.

LSRpools
Whenever CICS closes a file which is in an LSRPOOL, LSRPOOL statistics are collected (in addition to the file statistics) covering the period from the last interval. The following peak values are reset to the current value at each interval collection:

- Peak number of requests waiting for a string
- Maximum number of concurrent active file control strings.

The other statistics, which are not reset at an interval collection, cover the entire period from the time the LSRPOOL is created (when the first file is opened) until the LSRPOOL is deleted (when the last file is closed).

Programs
Whenever an installed program definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Transactions
Whenever an installed transaction definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Transaction classes
Whenever an installed transaction class definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Note: To ensure that accurate statistics are recorded USS statistics must be collected.

In particular, during a normal CICS shutdown, files are closed before the end of day statistics are gathered. This means that file and LSRPOOL end of day statistics will be zero, while the correct values will be recorded as unsolicited statistics.

Resetting statistics counters

When statistics are written to the SMF data set, the counters are reset in one of the following ways:

- Reset to zero
- Reset to 1
- Reset to current values (this applies to peak values)
- Are not reset
- Exceptions to the above.

For detailed information about the reset characteristics, see Appendix A, “CICS statistics tables” on page 329.

The arrival of the end-of-day time, as set by the ENDOFDAY parameters, always causes the current interval to be ended (possibly prematurely) and a new interval to be started. Only end-of-day statistics are collected at the end-of-day time, even if it coincides exactly with the expiry of an interval.

Changing the end-of-day value changes the times at which INTERVAL statistics are recorded immediately. In Figure 2, when the end-of-day is changed from midnight to 1700 just after 1400, the effect is for the interval times to be calculated from the new end-of-day time. Hence the new interval at 1500 as well as for the times after new end-of-day time.

When you change any of the INTERVAL values (and also when CICS is initialized), the length of the current (or first) interval is adjusted so that it expires after an integral number of intervals from the end-of-day time.

These rules are illustrated by the following example. **I** indicates an interval recording and **E** indicates an end-of-day recording.

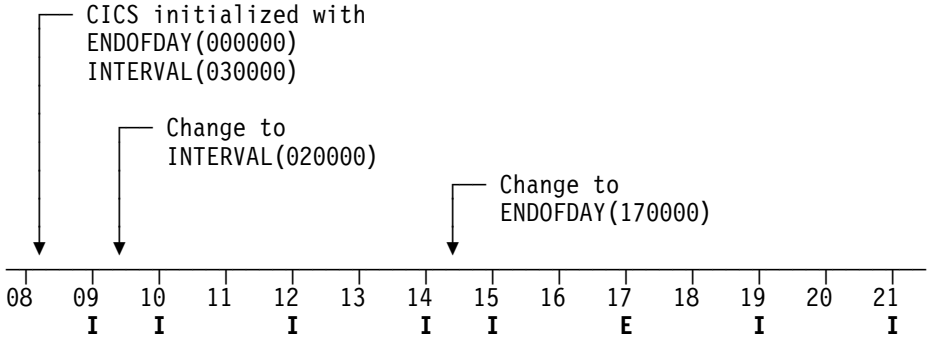


Figure 2. Resetting statistics counters

If you want your end-of-day recordings to cover 24 hours, set INTERVAL to 240000.

Note: Interval statistics are taken precisely on a minute boundary. Thus users with many CICS regions on a single MVS image could have every region writing statistics at the same time, if you have both the same interval and the same end of day period specified. This could cost up to several seconds of the entire CPU. If the cost becomes too noticeable, in terms of user response time around the interval expiry, you should consider staggering the intervals. One way of doing this while still maintaining very close correlation of intervals for all regions is to use a PLT program like the supplied sample DFH$STED which changes the end-of-day, and thus each interval expiry boundary, by a few seconds. See *CICS/ESA Operations and Utilities Guide* for further information about DFH$STED.

Processing CICS statistics

There are four ways of processing CICS statistics:

1. Use the CICS DFHSTUP offline utility. For guidance about retrieving CICS statistics from SMF, and about running DFHSTUP, see the *CICS/ESA Operations and Utilities Guide*.
2. Write your own program to report and analyze the statistics. For details about the statistics record types, see the assembler DSECTs named in each set of statistics. For programming information about the formats of CICS statistics SMF records, see the *CICS/ESA Customization Guide*.
3. Use the sample statistics program (DFH0STAT).

   You can use the statistics sample program, DFH0STAT, to help you determine and adjust the values needed for CICS/ESA storage parameters, for example, using DSALIM and EDSALIM. The program produces a report showing critical system parameters from the CICS/ESA dispatcher, an analysis of the CICS/ESA storage manager and loader statistics, and an overview of the MVS storage in use. The program demonstrates the use of the EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of a CICS/ESA system. You can use the sample program as provided or modify it to suit your needs. For more details, see Appendix B, “The sample statistics program, DFH0STAT” on page 441.
4. Use the Enterprise Performance Data Manager (EPDM) program to process CICS SMF records to produce joint reports with data from other SMF records. For more information, see Chapter 8, “Enterprise Performance Data Manager/MVS” on page 101.

Interpreting CICS statistics

In the following sections, as indicated in Table 1 on page 47, guidance is given to help with the interpretation of the statistics report. Information is presented in the order that it appears in the DFHSTUP report. Some headings have been omitted where they have little or no performance impact. Detailed information about the statistics tables is given in Appendix A, “CICS statistics tables” on page 329.

Table 1. Performance statistics types	
Statistic type	**page**
Dispatcher statistics	48
Dump statistics	53
Dynamic transaction backout statistics	53
Front end programming interface statistics	54
Files	54
ISC/IRC attach time statistics	63
Journals	55
Loader statistics	49
LSRPOOLS	55
Programs	54
Statistics domain statistics	47
Storage manager statistics	48
Task control	47
Temporary storage	49
Terminal autoinstall statistics	53
Terminals	56
Transaction class statistics	48
Transaction manager statistics	47
Transactions	53
Transient data (global)	50
Transient data (resource)	50
# User domain	50
VTAM statistics	51

Statistics domain statistics

Statistics recording on to an SMF data set can be a very CPU-intensive activity. The amount of activity depends more on the number of resources defined than the extent of their use. This may be another reason to maintain CICS definitions by removing redundant or over-allocated resources.

For more information about the statistics domain statistics, see page 401.

Transaction manager statistics

The “Times the MAXTASK limit reached” indicates whether MXT is constraining your system. The only time that you may need to constrain your system in this way is to reduce virtual storage usage. As most CICS virtual storage is above the 16MB line you may be able to run your system without MXT constraints, but note that CICS does preallocate storage, above and below the 16MB line, for each MXT whether or not it is used. Changing MXT affects your calculations for the dynamic

storage areas. see “Maximum task specification (MXT)” on page 259 for more information.

For more information about transaction manager statistics, see page 430.

Transaction class (TRANCLASS) statistics

If you are never at the limit of your transaction class setting then you might consider resetting its value, or review whether there is any need to continue specifying any transaction types with that class.

For more information, see the transaction class statistics on page 427.

Dispatcher statistics

TCB statistics

If the SIT parameter MNPER is set ON, “Accum CPU time/TCB” is the amount of CPU time which each CICS TCB consumed since the last time statistics were reset. Totalling the values of “Accum time in MVS wait” and “Accum time dispatched” gives you the approximate time since the last time CICS statistics were reset. The ratio of the “Accum CPU time /TCB” to this time shows the percentage usage of each CICS TCB. The “Accum CPU time/TCB” does not include uncaptured time, thus even a totally busy CICS TCB would be noticeably less than 100% busy from this calculation. If a CICS region is more than 70% busy by this calculation then you are approaching that region’s capacity.

Note: “Accum time dispatched” is NOT a measurement of CPU time because MVS can run higher priority work, for example, all I/O activity and higher priority regions, without CICS being aware.

For more information about dispatcher statistics, see page 339.

Storage manager statistics

Dynamic program compression releases programs which are not being used progressively as storage becomes shorter. However, short-on-storage conditions can still occur and are reported as “Times went short on storage.” If this value is not zero you might consider increasing the size of the dynamic storage area. Otherwise you should consider the use of MXT and transaction classes to constrain your system’s virtual storage.

Storage manager requests “Times request suspended,” and “Times cushion released,” indicate that storage stress situations have occurred. Some of which may not have produced a short-on-storage condition. For example, a GETMAIN request may cause the storage cushion to be released. However, loader can compress some programs, obtain the cushion storage, and avoid the short-on-storage condition.

Note: In the task subpools section, the “Current elem stg” is the number of bytes actually used while “Current page stg” is the number of pages containing one or more of these bytes.

For more information, see the CICS statistics tables on pages 402, 404, and 411.

Loader statistics

"Average loading time" = "Total loading time" / "Number of library load requests." This indicates the response time overhead suffered by tasks when accessing a program which has to be brought into storage. If "Average loading time" has increased over a period, consider MVS library lookaside usage. "Not-in-use" program storage is freed progressively so that the "Amount of the dynamic storage area occupied by not in use programs," and the free storage in the dynamic storage area are optimized for performance. Loader attempts to keep not-in-use programs in storage long enough to reduce the performance overhead of reloading the program. As the amount of free storage in the dynamic storage decreases, the not-in-use programs are freemained in order of those least frequently used to avoid a potential short-on-storage condition.

Note: The values reported are for the instant at which the statistics are gathered and vary since the last report.

"Average Not-In-Use queue membership time" = "Total Not-In-Use queue membership time" / "Number of programs removed by compression." This is an indication of how long a program is left in storage when not in use before being removed by the dynamic program storage compression (DPSC) mechanism. If the interval between uses of a program, that is, interval time divided by the number of times used in the interval, is less than this value, there is a high probability that the program is in storage already when it is next required.

Note: This factor is meaningful only if there has been a substantial degree of loader domain activity during the interval and may be distorted by startup usage patterns.

"Average suspend time" = "Total waiting time" / "Number of waited loader requests."

This is an indication of the response time impact which may be suffered by a task due to contention for loader domain resources.

Note: This calculation is not performed on requests that are currently waiting.

For more information, see the CICS statistics tables on page 374.

Temporary storage statistics

If a data item is written to temporary storage (using WRITEQ TS), a temporary storage queue is built. The temporary storage queue items are grouped into temporary storage groups (TSGIDs). The number of items per group that are allowed is controlled by the system initialization parameter, TSMGSET. The default is four. If the number of entries in the queue exceeds the limit specified by TSMGSET then a further TSGID will be allocated. This will be registered in TS statistics by "Queue extensions created."

The value of queue extensions created should be low compared with "times queues created." If it is high, consider increasing the TSMGSET system initialization parameter.

The "Writes more than control interval" is the number of writes of records whose length was greater than the control interval (CI) size of the TS data set. This value should be used to adjust the CI size. If the reported value is large, increase the CI

size. If the value is zero, consider reducing the CI size until a small value is reported.

The number of “times aux. storage exhausted” is the number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command, the use of RESP on the WRITEQ TS command, or WRITEQ TS NOSUSPEND command) may have been forced to abend. If this item appears in the statistics, increase the size of the temporary storage data set. “Buffer writes” is the number of WRITEs to the temporary storage data set. This includes both WRITEs necessitated by recovery requirements and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing buffer allocation using the system initialization parameter, TS=(b,s), where b is the number of buffers and s is the number of strings.

The “Peak number of strings in use” item is the peak number of concurrent I/O operations to the data set. If this is significantly less than the number of strings specified in the TS system initialization parameter, consider reducing the system initialization parameter to approach this number.

If the “Times string wait occurred” is not zero, then consider increasing the number of strings. For details about adjusting the size of the TS data set, the number of strings and buffers see the *CICS/ESA System Definition Guide*.

For more information, see the CICS statistics tables on page 414.

Transient data statistics

You should monitor the data provided by CICS on the amount of I/O activity for transient data, in the form of the number of READs and WRITEs to the transient data intrapartition data set. If there is a large amount of READ activity, this indicates that the buffer allocation may be insufficient, even though the “peak concurrent string access” may be fewer than the number allocated.

You should aim to minimize the “Intrapartition buffer waits” and “string waits” by increasing the number of buffers and the number of strings if you can afford any associated increase in your use of real storage.

For more information, see the CICS statistics tables on pages 432 and 435.

User domain statistics

The user domain attempts to minimize the number of times it calls the security
domain to create user security blocks (such as the ACEE), because this operation
is very expensive in both processor time and input/output operations. If possible,
each unique representation of a user is shared between multiple transactions. A
user-domain representation of a user can be shared if the following attributes are
identical:

- The userid.
- The groupid.
- The applid. This is not necessarily the same for all the users in a region. The
applid is shipped with the userid across MRO links.

- The port of entry. This can be the netname for users signed on at VTAM terminals, or the console name for users signed on at consoles. It is null for other terminal types and for users associated with non-terminal transactions.

The user domain keeps a count of the number of concurrent usages of a shared instance of a user. The count includes the number of times the instance has been associated with a CICS resource (such as a transient data queue) and the number of active transactions that are using the instance.

Whenever CICS adds a new user instance to the user domain, the domain attempts to locate that instance in its user directory. If the user instance already exists with the parameters described above, that instance is reused. USGDRRC records how many times this is done. However, if the user instance does not already exist, it needs to be added. This requires an invocation of the security domain and the external security manager. USGDRNFC records how many times this is necessary.

When the count associated with the instance is reduced to zero, the user instance is not immediately deleted: instead it is placed in a timeout queue controlled by the USRDELAY system initialization parameter. While it is in the timeout queue, the user instance is still eligible to be reused. If it is reused, it is removed from the timeout queue. USGTORC records how many times a user instance is reused while it was being timed out, and USGTOMRT records the average time that user instances remain on the timeout queue until they are removed.

However, if a user instance remains on the timeout queue for a full USRDELAY interval without being reused, it is deleted. USGTOEC records how many times this happens.

If USGTOEC is large compared to USGTORC, you should consider increasing the value of USRDELAY. But if USGTOMRT is much smaller than USRDELAY, you may be able to reduce USRDELAY without significant performance effect.

You should be aware that high values of USRDELAY may affect your security administrator's ability to change the authorities and attributes of CICS users, because those changes are not reflected in CICS until the user instance is refreshed in CICS by being flushed from the timeout queue after the USRDELAY interval. Some security administrators may require you to specify USRDELAY=0. This still allows some sharing of user instances if the usage count is never reduced to zero. Generally, however, remote users are flushed out immediately after the transaction they are executing has terminated, so that their user control blocks have to be reconstructed frequently. This results in poor performance. For more information, see “User domain statistics” on page 437.

VTAM statistics

The “peak RPLs posted” includes only the receive-any RPLs defined by the RAPOOL system initialization parameter. In non-HPO systems, the value shown can be larger than the value specified for RAPOOL, because CICS reissues each receive-any request as soon as the input message associated with the posted RPL has been disposed of. VTAM may well cause this reissued receive-any RPL to be posted during the current dispatch of terminal control. While this does not necessarily indicate a performance problem, a number much higher than the number of receive-any requests specified via RAPOOL may indicate, for MVS, that

VTAM was required to queue incoming messages in subpool 229 when no receive-any was available to accept the input. You should limit this VTAM queueing activity by providing a sufficient number of receive-any requests to handle all but the input message rate peaks.

In addition to indicating whether the value for the RAPOOL system initialization parameter is large enough, you can also use the “maximum number of RPLs posted” statistic (A03RPLX) to determine other information. This depends upon whether your MVS/ESA system has HPO or not.

For HPO, RAPOOL(A,B) allows the user to tune the active count (B). The size of the pool (A) should be dependent on the speed at which they get processed. The active count (B) has to be able to satisfy VTAM at any given time, and is dependent on the inbound message rate for receive-any requests.

Here is an example to illustrate the differences for an HPO and a non-HPO system. Suppose two similar CICS executions use a RAPOOL value of 2 for both runs. The number of RPLs posted in the MVS/HPO run is 2, while the MVS/non-HPO run is 31. This difference is better understood when we look at the next item in the statistics.

This item is not printed if the maximum number of RPLs posted is zero. In our example, let us say that the MVS/HPO system reached the maximum 495 times. The non-HPO MVS system reached the maximum of 31 only once. You might deduce from this that the pool is probably too small (RAPOOL=2) for the HPO system and it needs to be increased. An appreciable increase in the RAPOOL value, from 2 to, say, 6 or more, should be tried. As you can see from the example given below, the RAPOOL value was increased to 8 and the maximum was reached only 16 times:

```
MAXIMUM NUMBER OF RPLS POSTED         8
NUMBER OF TIMES REACHED MAXIMUM      16
```

In a non-HPO system, these two statistics are less useful, except that, if the maximum number of RPLs posted is less than RAPOOL, RAPOOL can be reduced, thereby saving virtual storage.

VTAM SOS simply means that a CICS request for service from VTAM was rejected with a VTAM sense code indicating that VTAM was unable to acquire the storage required to service the request. VTAM does not give any further information to CICS, such as what storage it was unable to acquire.

This situation most commonly arises at network startup or shutdown when CICS is trying to schedule requests concurrently, to a larger number of terminals than during normal execution. If the count is not very high, it is probably not worth tracking down. In any case, CICS automatically retries the failing requests later on.

If your network is growing, however, you should monitor this statistic and, if the count is starting to increase, you should take action. Use D NET,BFRUSE to check if VTAM is short on storage in its own region and increase VTAM allocations accordingly if this is required.

The maximum value for this statistic is 99, at which time a message is sent to the console and the counter is reset to zero. However, VTAM controls its own buffers and gives you a facility to monitor buffer usage.

If you feel that D NET,BFRUSE is insufficient, you can activate SMS tracing in VTAM to sample buffer activity at regular intervals. If you have installed NetView, you can also have dynamic displays of the data that is obtained with D NET, BFRUSE.

For more information, see the CICS statistics tables on page 438.

Terminal autoinstall statistics

If "times SETLOGON HOLD issued" is not zero, your region is reaching your AIQMAX limit. If this statistic increases over a period of time, you may need to review whether you can afford to increase AIQMAX.

For more information, see the CICS statistics tables on page 419.

Dump statistics

Both transaction and system dumps are very expensive and should be thoroughly investigated and eliminated.

For more information, see the CICS statistics tables on page 343.

Dynamic transaction backout statistics

Normally, dynamic log records are simply added to a main storage dynamic log buffer. If this proves to be too small, records are spilled to a main storage chain. There is a performance cost in spilling records, so the dynamic log buffer size is automatically tuned to keep this number low compared with the number of records written.

If the ratio of records spilled to records logged is too high, it may be that the value of the system initialization parameter DBUFSZ is too low. The DBUFSZ value imposes an upper limit on the dynamic log buffer size that may be allocated by the self-tuning mechanism.

Because the initial allocations are half the maximum, the dynamic self-tuning mechanism may take some time to settle down.

If the "number of records spilled" is consistently over 2% of the "number of records logged," you should increase the buffer size.

For more information, see the CICS statistics tables on page 348.

Transaction statistics

Use these statistics to find out which transactions (if any) had storage violations.

It is also possible to use these statistics for capacity planning purposes. But remember, many systems experience both increasing cost per transaction as well as increasing transaction rate.

For more information, see the CICS statistics tables on page 425.

Program statistics

"Average fetch time" is an indication of how long it actually takes MVS to perform a load from the partitioned data set in the RPL concatenation into CICS managed storage.

"RPL offset" is the average for each RPL offset of: "Program size" / "Average fetch time." This is an indication of the byte transfer rate during loads from a particular partitioned data set. A comparison of these values may assist you to detect bad channel loading or file layout problems.

For more information, see the CICS statistics tables on page 397.

Front end programming interface (FEPI) statistics

CICS monitoring and statistics data can be used to help tune FEPI applications, and to control the resources that they use. FEPI statistics contain data about the use of each FEPI pool, a particular target in a pool, and each FEPI connection.

For more information, see the CICS statistics tables on page 348.

File statistics

File statistics collect data about the number of application requests against your data sets. They indicate the number of requests for each type of service that are processed against each file. If the number of requests is totalled daily or for every CICS execution, the activity for each file can be monitored for any changes that occur. Note that these file statistics may have been reset during the day; to obtain a figure of total activity against a particular file during the day, refer to the DFHSTUP summary report. Other data pertaining to file statistics and special processing conditions are also collected.

The wait-on-string number is only significant for files related to VSAM data sets. For VSAM, STRNO=5 in the file definition means, for example, that CICS permits five concurrent requests to this file. If a transaction issues a sixth request for the same file, this request must wait until one of the other five requests has completed ("wait-on-string").

The number of strings associated with a file when specified through resource definition online.

String number setting is important for performance. Too low a value causes excessive waiting for strings by tasks and long response times. Too high a value increases VSAM virtual storage requirements and therefore real storage usage. However, as both virtual storage and real storage are above the 16MB line, this may not be a problem. In general, the number of strings should be chosen to give near zero "wait on string" count.

Note: Increasing the number of strings can increase the risk of deadlocks because of greater transaction concurrency. To minimize the risk you should ensure that applications follow the standards set in the *CICS/ESA Application Programming Guide*.

A file can also “wait-on-string” for an LSRpool string. This type of wait is reflected in the local shared resource pool statistics section (see “LSRPOOL statistics” on page 55) and not in the file wait-on-string statistics.

If you are using data tables, an extra line appears in the DFHSTUP report for those files defined as data tables. “Read requests”, “Source reads”, and “Storage alloc(K)” are usually the numbers of most significance. For a CICS-maintained table a comparison of the difference between “read requests” and “source reads” with the total request activity reported in the preceding line shows how the request traffic divides between using the table and using VSAM and thus indicates the effectiveness of converting the file to a CMT. “Storage alloc(K)” is the total storage allocated for the table and provides guidance to the cost of the table in storage resource, bearing in mind the possibility of reducing LSRpool sizes in the light of reduced VSAM accesses.

When a file is closed, and unsolicited statistics (USS) record containing file statistics, mapped by the DFHA17DS DSECT, is written to SMF. If the file is in an LSRPOOL, a USS record containing LSRPOOL file statistics is also written, which can be mapped by the DFHA09DS DSECT.

For more information, see the CICS statistics tables on page 352.

Journal statistics

Each journal employs two buffers for efficiency; CICS can thus use one buffer for output, while concurrently receiving records from transactions in the other buffer. If the receiving buffer becomes full before output to the other buffer has completed, the “buffer full” count is incremented by one. This situation causes significant increase in internal response time as every transaction attempting to put a record into the journal has to wait. The buffer size should be increased to reduce this problem.

For more information, see the CICS statistics tables on page 372.

LSRPOOL statistics

CICS supports the use of up to eight LSRpools. CICS produces two sets of statistics for LSRpool activity: one set detailing the activity for each LSRpool, and one set giving details for each file associated with an LSRpool. Statistics are printed for all pools that have been built (a pool is built when at least one file using the pool has been opened).

As each file in an LSRPOOL is closed, an unsolicited statistics record containing LSRPOOL file statistics is written to SMF. This can be mapped by the DFHA09DS DSECT.

You should usually aim to have no requests that waited for a string. If you do then the use of MXT may be more effective.

When the last open file in an LSRPOOL is closed, the pool is deleted. The subsequent unsolicited statistics (USS) LSRPOOL record written to SMF can be mapped by the DFHA08DS DSECT.

The fields relating to the size and characteristics of the pool (maximum key length, number of strings, number and size of buffers) may be those which you have specified for the pool, through resource definition online command DEFINE LSRPOOL. Alternatively, if some, or all, of the fields were not specified, the values of the unspecified fields are those calculated by CICS when the pool is built.

It is possible to change the LSRPOOL specification of a file when it is closed, but you must then consider the characteristics of the pool that the file is to share if the pool is already built, or the file open may fail. If the pool is not built and the pool characteristics are specified by you, take care that these are adequate for the file. If the pool is not built and CICS calculates all or some of the operands, it may build the pool creations of that pool. The statistics show all creations of the pool, so any changed characteristics are visible.

You should consider specifying separate data and index buffers if you have not already done so. This is especially true if index CI sizes are the same as data CI sizes.

You should also consider using Hiperspace buffers while retaining a reasonable number of address space buffers. Hiperspace buffers tend to give CPU savings of keeping data in memory, exploiting the relatively cheap expanded storage, while allowing central storage to be used more effectively.

For more information, see the CICS statistics tables on pages 385.

Terminal statistics

There are a number of ways in which terminal statistics are important for performance analysis. From them, you can get the number of inputs and outputs, that is, the loading of the system by end users. Line-transmission faults and transaction faults are shown (these both have a negative influence on performance behavior).

For more information, see the CICS statistics tables on page 422.

ISC/IRC system and mode entry statistics

You can use the ISC/IRC system and mode entry statistics to detect some problems in a CICS intersystem environment.

The following section attempts to identify the kind of questions you may have in connection with system performance, and describes how answers to those questions can be derived from the statistics report. It also describes what actions, if any, you can take to resolve ISC/IRC performance problems.

Some of the questions you may be seeking an answer to when looking at these statistics are these:

- Are there enough sessions defined?
- Is the balance of **contention winners** to **contention losers** correct?
- Is there conflicting usage of APPC modegroups?
- What can be done if there are unusually high numbers, compared with normal or expected numbers, in the statistics report?

Summary connection type for statistics fields

The following two tables show the connection type that is relevant for each statistics field:

Table 2. ISC/IRC system entries

System entry	Field	IRC	LU6.1	APPC
Connection name	A14CNTN	X	X	X
AIDS in chain	A14EALL	X	X	X
Generic AIDS in chain	A14ESALL	X	X	X
ATIs satisfied by contention losers	A14ES1		X	
ATIs satisfied by contention winners	A14ES2	X	X	
Peak contention losers	A14E1HWM	X	X	
Peak contention winners	A14E2HWM	X	X	
Peak outstanding allocates	A14ESTAM	X	X	X
Total number of allocates	A14ESTAS	X	X	X
Queued allocates	A14ESTAQ	X	X	X
Failed link allocates	A14ESTAF	X	X	X
Failed allocates due to sessions in use	A14ESTAO	X	X	X
Total bids sent	A14ESBID		X	
Current bids in progress	A14EBID		X	
Peak bids in progress	A14EBHWM		X	
File control function shipping requests	A14ESTFC	X	X	X
Interval control function shipping requests	A14ESTIC	X	X	X
TD function shipping requests	A14ESTTD	X	X	X
TS function shipping requests	A14ESTTS	X	X	X
DLI function shipping requests	A14ESTDL	X	X	X
Terminal sharing requests	A14ESTTC	X		X

All the fields below are specific to the mode group of the mode name given.

Table 3 (Page 1 of 2). ISC/IRC mode entries

Mode entry	Field	IRC	LU6.1	APPC
Mode name	A20MODE			X
ATIs satisfied by contention losers	A20ES1			X
ATIs satisfied by contention winners	A20ES2			X
Peak contention losers	A20E1HWM			X
Peak contention winners	A20E2HWM			X
Peak outstanding allocates	A20ESTAM			X
Total specific allocate requests	A20ESTAS			X
Total specific allocates satisfied	A20ESTAP			X

Table 3 (Page 2 of 2). ISC/IRC mode entries				
Mode entry	**Field**	**IRC**	**LU6.1**	**APPC**
Total generic allocates satisfied	A20ESTAG			X
Queued allocates	A20ESTAQ			X
Failed link allocates	A20ESTAF			X
Failed allocates due to sessions in use	A20ESTAO			X
Total bids sent	A20ESBID			X
Current bids in progress	A20EBID			X
Peak bids in progress	A20EBHWM			X

For more information about the usage of individual fields, see the CICS statistics described under “ISC/IRC system and mode entries” on page 361.

General guidance for interpreting ISC/IRC statistics

You should read the following notes on how the information about interpreting the ISC/IRC statistics is presented:

1. Usage of A14xxx and A20xxx fields:
   - In most cases, the guidance given in the following section relates to all connection types, that is, IRC, LU6.1, and APPC. Where the guidance is different for a particular connection type, the text indicates the relevant type of connection.
   - The statistics fields which relate to IRC and LU6.1 are always prefixed A14, whereas the APPC fields can be prefixed by A14 or A20. For more information on which field relates to which connection type, see Table 2 on page 57 and Table 3 on page 57.
2. Use of the terms “Contention Winner” and “Contention Loser”:
   - APPC sessions are referred to as either **contention winners** or **contention losers**. These are equivalent to secondaries (SEND sessions) and primaries (RECEIVE sessions) when referring to LU6.1 and IRC.
3. Tuning the number of sessions defined:
   - In the following sections it is sometimes stated that if certain counts are too high then you should consider making more sessions available. In these cases, be aware that as the number of sessions defined in the system is increased, it may have the following effects:
     – Increased use of real and virtual storage.
     – Increased use of storage on GATEWAY NCPs in network.
     – Increased use of storage by VTAM.
     – Increased line loading in the network.
     – The back-end CICS system (AOR) may not be able to cope with the increased workload from the TOR.
     – Possible performance degradation due to increased control block scanning by CICS.
   - The recommendation is to set the number of sessions available to the highest value you think you may need, and then through monitoring the statistics (both ISC/IRC and terminal statistics) over a number of CICS

runs, reduce the number of sessions available to just above the number required to avoid problems.

4. Tuning the number of contention winner and contention loser sessions available:
   - Look at both sides of the connection when carrying out any tuning, as changing the loading on one side could inversely affect the other. Any change made to the number of contention winner sessions available in the TOR has an effect on the number of contention loser sessions in the AOR.
5. Establish a connection profile for comparison and measurement.

   One of the objectives of a tuning exercise should be to establish a profile of the usage of CICS connections during both normal and peak periods. Such usage profiles can then be used as a reference point when analyzing statistics to help you:
   - Determine changed usage patterns over a period of time
   - To anticipate potential performance problems before they become critical.

Are there enough sessions defined?

To help you determine whether you have enough sessions defined, you can check a number of peak fields that CICS provides in the statistics report.
These are:

1. **"Peak outstanding allocates"** (fields A14ESTAM and A20ESTAM)
   "Total number of allocates" (field A14ESTAS)
   "Total specific allocate requests" (field A20ESTAS).

   When reviewing the number of sessions for APPC modegroups, and the number of "Peak outstanding allocates" appears high in relation to the "Total number of allocates," or the "Total specific allocate requests" within a statistics reporting period, it could indicate that the total number of sessions defined is too low.
2. **"Peak contention winners"** (fields A14E2HWM and A20E2HWM)
   "Peak contention losers" (fields A14E1HWM and A20E1HWM)

   If the number of ("Peak contention winners" + "Peak contention losers") equals the maximum number of sessions available (as defined in the SESSIONS definition), this indicates that, at some point in the statistics reporting period, all the sessions available were, potentially, in use. While these facts alone may not indicate a problem, if CICS also queued or rejected some allocate requests during the same period, then the total number of sessions defined is too low.

Action: Consider making more sessions available with which to satisfy the allocate requests. Enabling CICS to satisfy allocate requests without the need for queueing may lead to improved performance.

However, be aware that increasing the number of sessions available on the front-end potentially increases the workload to the back-end, and you should investigate whether this is likely to cause a problem.

Is the balance of contention winners to contention losers correct?

There are several ways to determine the answer to this, as CICS provides a number of fields which show contention winner and contention loser usage.

The following fields should give some guidance as to whether you need to increase the number of contention winner sessions defined:

1. **"Current bids in progress"** (fields A14EBID and A20EBID)
   "Peak bids in progress" (fields A14EBHWM and A20EBHWM)

   The value "Peak bids in progress" records the maximum number of bids in progress at any one time during the statistics reporting period. "Current bids in progress" is always less than or equal to the "Peak bids in progress."

   Ideally, these fields should be kept to zero. If either of these fields is high, it indicates that CICS is having to perform a large number of bids for contention loser sessions.

2. **"Failed allocates due to sessions in use"** (fields A14ESTAO and A20ESTAO)

   This value is incremented for allocates that are rejected with a SYSBUSY response because no sessions are immediately available (that is, for allocate requests with the NOSUSPEND or NOQUEUE option specified). This value is also incremented for allocates that are queued and then rejected with an AAL1 abend code; the AAL1 code indicates the allocate is rejected because no session became available within the specified deadlock timeout (DTIMOUT) time limit.

   If the number of "Failed allocates due to sessions in use" is high within a statistics reporting period, it indicates that not enough sessions were immediately available, or available within a reasonable time limit.

3. **"Peak contention losers"** (fields A14E1HWM and A20E1HWM).

   If the number of "Peak contention losers" is equal to the number of contention loser sessions available, then the number of contention loser sessions defined may be too low. Alternatively, for APPC/LU6.1, CICS could be using the contention loser sessions to satisfy allocates due to a lack of contention winner sessions. This should be tuned at the front-end in conjunction with winners at the back-end. For information on how to specify the maximum number of sessions, and the number of contention winners, see the *CICS/ESA Resource Definition Guide*.

Actions:

For APPC, consider making more contention winner sessions available, which should reduce the need to use contention loser sessions to satisfy allocate requests and, as a result, should also make more contention loser sessions available.

For LU6.1, consider making more SEND sessions available, which decreases the need for LU6.1 to use primaries (RECEIVE sessions) to satisfy allocate requests.

For IRC, there is no bidding involved, as MRO can never use RECEIVE sessions to satisfy allocate requests. If "Peak contention losers (RECEIVE)" is equal to the number of contention loser (RECEIVE) sessions on an IRC link, the number of allocates from the remote system is possibly higher than the receiving system can

cope with. In this situation, consider increasing the number of RECEIVE sessions available.

Note: The usage of sessions depends on the direction of flow of work. Any tuning which increases the number of winners available at the front-end should also take into account whether this is appropriate for the direction of flow of work over a whole period, such as a day, week, or month.

Is there conflicting usage of APPC modegroups?

There is a possibility of conflicting APPC modegroup usage, where a mixture of generic and specific allocate requests is used within a CICS region.

A specific allocate is an allocate request that specifies a particular (specific) mode group of sessions to allocate from, whereas a generic allocate does not specify any particular mode group only the system to which an allocate is required. In the latter case CICS determines the session and mode group to allocate.

The fields you need to investigate to answer this question, are:

"Total generic allocates satisfied" (field A20ESTAG)
"Total specific allocate requests" (field A20ESTAS)
"Peak outstanding allocates" (field A20ESTAM)
"Total specific allocates satisfied" (field A20ESTAP).

If the "Total generic allocates satisfied" is much greater than "Total specific allocate requests," and "Peak outstanding allocates" is not zero, it could indicate that generic allocates are being made only, or mainly, to the first modegroup for a connection.

This could cause a problem for any specific allocate, because CICS initially tries to satisfy a generic allocate from the first modegroup before trying other modegroups in sequence.

Action: Consider changing the order of the installed modegroup entries. Modegroups for a connection are represented by TCT mode entries (TCTMEs), with the modegroup name being taken from the MODENAME specified on the SESSIONS definition. The order of the TCTMEs is determined by the order in which CICS installs the SESSIONS definitions, which is in the order of the SESSIONS name as stored on the CSD (ascending alphanumeric key sequence). See Figure 3 on page 62 for an illustration of this. To change the order of the TCTMEs, you must change the names of the SESSIONS definitions. You can use the CEDA RENAME command with the AS option to rename the definition with a different SESSIONS name within the CSD group. By managing the order in which the TCTMEs are created you can ensure that specific allocates reference modegroups lower down the TCTME chain, and avoid conflict with the generic ALLOCATEs. **Alternatively, make all allocates specific allocates**.

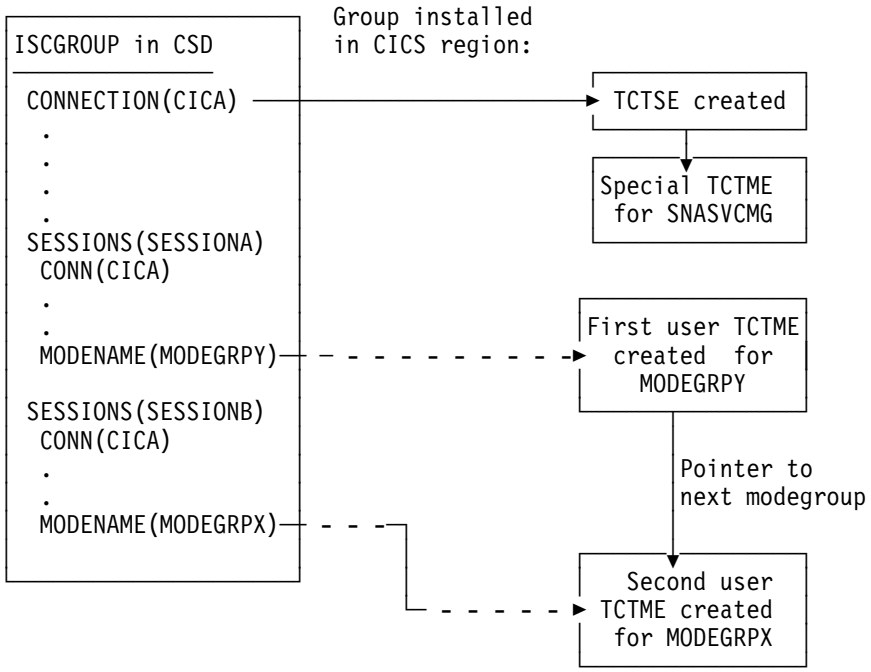


Figure 3. How the sequence of TCT mode entries is determined

What if there are unusually high numbers in the statistics report?

When looking down the **ISC/IRC system and mode entries** statistics report, you may notice a number of fields that appear to be unusually high in relation to all others. This section lists some of those fields, and what action you can take to reduce their numbers:

1. **"Peak contention losers"** (fields A14E1HWM and A20E1HWM).

   If the number of "Peak contention losers" is equal to the number of contention loser sessions available, then the number of contention loser sessions defined may be too low, or, if your links are APPC/LU6.1, CICS could be using the contention loser sessions to satisfy allocates due to a lack of contention winner sessions.

   Action: Consider making more contention winner sessions available with which to satisfy the allocate requests.

2. **"Peak outstanding allocates"** (fields A14ESTAM and A20ESTAM)

   If the number of "Peak outstanding allocates" appears high, in relation to the "Total number of allocates," or the "Total specific allocate requests" for APPC modegroups within a statistics reporting period, it could indicate that the total number of sessions defined is too low, or that the remote system cannot cope with the amount of work being sent to it.

   Action: Consider making more sessions available with which to satisfy the allocate requests, or reduce the number of allocates being made.

3. **"Failed link allocates"** (fields A14ESTAF and A20ESTAF)

   If this value is high within a statistics reporting period, it indicates something was wrong with the state of the connection. The most likely cause is either that the connection is released, out of service, or it has a closed mode group.

Action: Examine the state of the connection that CICS is trying to allocate a session on, and resolve any problem that is causing the allocates to fail.

To help you to resolve a connection failure, check the CSMT log for the same period covered by the statistics for any indication of problems with the connection that the statistics relate to.

It may also be worth considering writing a connection status monitoring program, which can run in the background and regularly check connection status and take remedial action to re-acquire a released connection. This may help to minimize outage time caused by connections being unavailable for use. See the *CICS/ESA System Programming Reference* manual for programming information about the EXEC CICS INQUIRE|SET CONNECTION and the EXEC CICS INQUIRE|SET MODENAME commands that you would use in such a program.

4. **"Failed allocates due to sessions in use"** (fields A14ESTAO and A20ESTAO)

   This value is incremented for allocates that have been rejected with a SYSBUSY response because no sessions were immediately available, and the allocate requests were made with the NOSUSPEND or NOQUEUE option specified. This value is also incremented for allocates that have been queued and then rejected with an AAL1 abend code; the AAL1 code indicates the allocate was rejected because no session was available within the specified deadlock timeout (DTIMOUT) time limit.

   If the number of "Failed allocates due to sessions in use" is high, within a statistics reporting period, it indicates that not enough sessions were immediately available, or available within a reasonable time limit.

   Action: The action is to consider making more contention winner sessions available. This action would result in a reduction in the amount of bidding being carried out, and the subsequent usage of contention loser sessions.

5. **"Peak bids in progress"** (fields A14EBHWM and A20EBHWM)

   Ideally, these fields should be kept to zero. If either of these fields are high, it indicates that CICS is having to perform a large amount of bidding for sessions.

   Action: Consider making more contention winner sessions available, to satisfy allocate requests.

ISC/IRC attach time entries

ISC/IRC Signon activity. Specify USRDELAY=n, where n is the maximum time, in the range 0 through 10080 (up to 7 days), that an eligible userid and its associated attributes are to be retained in the user table if the userid is unused. If the number of "entries reused" in signon activity is low, and the "entries timed out" value for signon activity is high, then the value of the USRDELAY system initialization parameter should be increased. The "average reuse time between entries" gives some indication of the time that could be used for the USRDELAY system initialization parameter. Specifying a low value for USRDELAY results in being less able to reuse the entry in the user table. If the value is too low, the increased number of security rebuilds considerably increases the CPU utilization.

ISC Persistent verification (PV) activity. If the number of "entries reused" in the PV activity is low, and the "entries timed out" value is high, then the PVDELAY

system initialization parameter should be increased. The “average reuse time between entries” gives some indication of the time that could be used for the PVDELAY system initialization parameter.

Note: If there are a lot of either signed-on or PV-entries timed out, and not many reused, your performance may be degraded because of the need to make calls to an external security manager, such as RACF for security checking.

For more information, see the CICS statistics tables on page 371.

Chapter 6. The CICS monitoring facility

This chapter is divided as follows:

- An introduction to CICS monitoring
- The classes of monitoring data
- Methods used to process monitoring data.

Introduction to CICS monitoring

CICS monitoring collects data about the performance of all user- and CICS-supplied transactions during online processing for later offline analysis. The records produced by CICS monitoring are of the MVS System Management Facility (SMF) type 110, and are written to an SMF data set.

Note: Statistics records and some journaling records are also written to the SMF data set as type 110 records. You might find it particularly useful to process the statistics records and the monitoring records together, because statistics provide resource and system information that is complementary to the transaction data produced by CICS monitoring. The contents of the statistics fields, and the procedure for processing them, are described in Appendix A, “CICS statistics tables” on page 329.

Monitoring data is useful both for performance tuning and for charging your users for the resources they use.

The classes of monitoring data

Three types, or “classes”, of monitoring data may be collected. These are **performance** class data, **exception** class data, and **SYSEVENT** data.

Performance class data

Performance class data is detailed transaction-level information, such as the processor and elapsed time for a transaction, or the time spent waiting for I/O. At least one performance record is written for each transaction that is being monitored.

Performance class data provides detailed, resource-level data that can be used for accounting, performance analysis, and capacity planning. This data contains information relating to individual task resource usage, and is completed for each task when the task terminates.

You can enable performance-class monitoring by coding MNPER=ON (together with MN=ON) in the system initialization table (SIT). Alternatively you can use either the (CEMT SET MONITOR ON PERF) or EXEC CICS SET MONITOR STATUS(ON) PERFCLASS(PERF) commands.

This information could be used periodically to calculate the charges applicable to different tasks. If you want to set up algorithms for charging users for resources used by them, you could use this class of data collection to update the charging information in your organization's accounting programs. (For previous versions of CICS, we did not recommend charging primarily on exact resource usage, because of the overheads involved in getting these figures.)

Exception class data

Exception class data is information on exceptional conditions suffered by a transaction. This data highlights possible problems in system operation. There is one exception record for each exception condition. Exception records are produced after each of the following conditions encountered by a transaction has been resolved:

- Wait for storage in the CDSA
- Wait for storage in the UDSA
- Wait for storage in the SDSA
- Wait for storage in the RDSA
- Wait for storage in the ECDSA
- Wait for storage in the EUDSA
- Wait for storage in the ESDSA
- Wait for storage in the ERDSA
- Wait for auxiliary temporary storage
- Wait for auxiliary temporary storage string
- Wait for auxiliary temporary storage buffer
- Wait for file string
- Wait for file buffer
- Wait for LSRPOOL string.

An exception record is created each time any of the resources covered by exception class monitoring becomes constrained by system bottlenecks. If performance data is also being recorded, it keeps a count of the number of exception records generated for each task. The exception records can be linked to the performance data by the transaction identifier in both records.

This data is intended to help you identify constraints that affect the performance of your transaction. The information is written to a SMF data set as soon as the task that was originally constrained has been released.

You can enable exception-class monitoring by coding MNEXC=ON (together with MN=ON) in the SIT. Alternatively you can use, either the CEMT command (CEMT SET MONITOR ON EXCEPT) or EXEC CICS SET MONITOR STATUS(ON) EXCEPTCLASS(EXCEPT).

The SYSEVENT class of monitoring data

SYSEVENT data is a special kind of transaction timing information.

CICS invokes the MVS System Resource Manager (SRM) macro SYSEVENT at the end of every transaction to record the elapsed time of the transaction.

You can enable SYSEVENT class monitoring by coding MNEVE=ON (together with MN=ON) in the SIT. Alternatively you can use, either the CEMT command (CEMT SET MONITOR ON EVENT) or EXEC CICS SET MONITOR STATUS(ON) EVENTCLASS(EVENT).

If the SYSEVENT option is used, at the end of each transaction CICS issues a Type 55 (X'37') SYSEVENT macro. This records each transaction ID, the associated terminal ID, and the elapsed time duration of each transaction. This information is collected by the SRM and output, depending on the Resource Measurement Facility (RMF) options set, can be written to SMF data sets.

CICS/ESA Monitoriing Facility (CMF) and the MVS workload manager

If you are running CICS/ESA 4.1 in an MVS/ESA 5.1 workload management environment in **compatibility mode**, CICS/ESA supports the SYSEVENT class of monitoring by default, regardless of the status of the monitoring options. You do not need to set monitoring and sysevent monitoring on (with MN=ON and MNEVE=ON respectively).

If you are running CICS/ESA 4.1 in an MVS/ESA 5.1 workload management environment in **goal mode**, the MVS workload manager provides transaction activity report reporting which replaces the SYSEVENT class of monitoring.

Using CICS monitoring SYSEVENT information with RMF

This section explains how to use the CICS monitoring facility with the Resource Measurement Facility (RMF) to obtain transaction rate reporting.

CICS usage of RMF transaction reporting

CICS monitoring facility with RMF provides a very useful tool for performing day-to-day monitoring of CICS transaction rates and response times.

The objective of using the CICS monitoring facility with RMF is to enable transaction rates and internal response times to be monitored without incurring the overhead of running the full CICS monitoring facility and associated reporting. This approach may be useful when only transaction statistics are required, rather than the very detailed information that CICS monitoring facility produces. An example of this is the monitoring of a production system where the minimum overhead is required.

CICS monitoring facility use of SYSEVENT

The CICS monitoring facility issues a SYSEVENT macro that gives to MVS/SRM the following information:

- The time at which the user-task was attached.
- The subsystem identification. This is derived from the first four characters of the CICS generic APPLID or from the four character name specified on the MNSUBSYS parameter if it is specified in the system initialization table (SIT).
- The transaction identifier of the task. This is the name of the CICS RDO transaction in the CICS program control table. This can be the name of a CICS system transaction, such as CSMI, CSNC, or CSPG.
- The user identifier.
- The specific APPLID of the CICS region. This is derived from the system initialization parameter, APPLID. It is expressed as the full 8 bytes of the transaction class parameter.

MVS IEAICS member

An IEAICS member needs to be coded and placed in SYS1.PARMLIB on the MVS system. For further information about this, see the *MVS/ESA Initialization and Tuning Guide*. Reporting groups are assigned to the CICS system as a whole and to individual transactions.

How CMF SYSEVENT data is mapped to the IEAICSxx member of SYS1.PARMLIB

Table 4. How CMF SYSEVENT data is mapped to IEAICSxx

SYSEVENT macro	IEAICSxx member	CICS monitoring facility data
Transaction start time	N/A	Time at which user-task attached
Subsystem name	SUBSYS=	First 4 characters of the Generic APPLID
Transaction name	TRXNAME=	Transaction ID of task
User identification	USERID=	User ID
Transaction class	TRXCLASS=	The specific APPLID of the CICS region

For more information about how to use RMF, refer to the *MVS/ESA Resource Measurement Facility (RMF), Version 4.1.1 – Monitor I & II Reference and Users Guide*. If records are directed to SMF, refer to the *MVS/ESA System Management Facilities* manual. The following example shows the additional parameters that you need to add to your IEAICS member for two MRO CICS systems:

```
SUBSYS=ACIC,RPGN=100       /* CICS SYSTEM ACIC HAS REPORTING    */
  TRXNAME=CEMT,RPGN=101    /* GROUP OF 100 AND THERE ARE        */
  TRXNAME=USER,RPGN=102    /* THREE INDIVIDUAL GROUPS FOR       */
  TRXNAME=CSMI,RPGN=103    /* SEPARATE TRANSACTIONS             */
SUBSYS=BCIC,RPGN=200       /* CICS SYSTEM BCIC HAS REPORTING    */
  TRXNAME=CEMT,RPGN=201    /* GROUP OF 200 AND THERE ARE        */
  TRXNAME=USER,RPGN=202    /* THREE INDIVIDUAL GROUPS FOR       */
  TRXNAME=CSMI,RPGN=203    /* SEPARATE TRANSACTIONS             */
```

Notes:

1. The reporting group (number 100) assigned to the ACIC subsystem reports on all transactions in that system.
2. RMF reports on an individual transaction by name only if it is assigned a unique reporting group. If multiple transactions are defined with one reporting group, the name field is left blank in the RMF reports.

ERBRMF member for Monitor I session

This member defines the options that are used on the RMF Monitor I background session. This session does not include transaction reporting as used by CICS, but a Monitor I session has first to be active. A WKLD has to be defined to allow TRX reporting to be activated.

ERBRMF member for Monitor II session

This member defines the options that are used on the RMF Monitor II background session. This session performs transaction reporting as used by CICS. TRX defaults to TRX(ALLPGN) which reports on all transactions. Individual transactions can be named if desired.

RMF operations

A RMF job has to be started and this includes the Monitor I session. The RMF job should be started before initializing CICS. The RMF Monitor II session is started by the command F RMF,S aa,MEMBER(xx) where 'aa' indicates alphabetic characters and 'xx' indicates alphanumeric characters.

Using the CICS monitoring facility with EPDM

Enterprise Performance Data Manager (EPDM) assists you in performance management and service-level management of a number of IBM products. EPDM's CICS Performance feature provides reports for your use in analysing the performance of CICS/ESA and CICS/MVS. See Chapter 8, "Enterprise Performance Data Manager/MVS" on page 101 for more information.

Event monitoring points

Product-Sensitive programming interface

CICS monitoring data is collected at system-defined event monitoring points (EMPs) in the CICS code. Although you cannot relocate these monitoring points, you can choose which **classes** of monitoring data you want to be collected. Programming information about CICS monitoring is in the *CICS/ESA Customization Guide*.

If you want to gather more performance class data than is provided at the system-defined event monitoring points, you can code additional EMPs in your application programs. At these points, you can add or change up to 16384 bytes of user data in each performance record. Up to this maximum of 16384 bytes you can have, for each ENTRYNAME qualifier, any combination of the following:

- Between 0 and 256 counters
- Between 0 and 256 clocks
- A single 8192-byte character string.

You could use these additional EMPs to count the number of times a certain event occurs, or to time the interval between two events. If the performance class was active when a transaction was started, but was not active when a user EMP was issued, the operations defined in that user EMP would still execute on that transaction's monitoring area. The DELIVER option would result in a loss of data at this point, because the generated performance record cannot be output while the performance class is not active. If the performance class was not active when a transaction was started, then the user EMP would have no effect.

User EMPs can use the EXEC CICS MONITOR command. For programming information about this command, refer to the *CICS/ESA Application Programming Reference* manual.

Additional EMPs are provided in some IBM program products, such as DL/I. From CICS's point of view, these are like any other user-defined EMP. EMPs in user applications and in IBM program products are identified by a decimal number. The numbers 1 through 199 are available for EMPs in user applications, and the numbers from 200 through 255 are for use in IBM program products. The numbers can be qualified with an 'entryname', so that you can use each number more than once. For example, PROGA.1, PROGB.1, and PROGC.1, identify three different EMPs because they have different entrynames.

For each user-defined EMP there must be a corresponding monitoring control table (MCT) entry, which has the same identification number and entryname as the EMP that it describes.

You do not have to assign entrynames and numbers to system-defined EMPs, and you do not have to code MCT entries for them.

Here are some ideas about how you might make use of the CICS and user fields provided with the CICS monitoring facility:

- If you want to time how long it takes to do a table lookup routine within an application, then you code an EMP with, say, ID=50 just before the table lookup routine and an EMP with ID=51 just after the routine. The system programmer codes a TYPE=EMP operand in the MCT for ID=50 to start user clock 1. You also code a TYPE=EMP operand for ID=51 to stop user clock 1. The application executes. When EMP 50 is processed, user clock 1 is started. When EMP 51 is processed, the clock is stopped.
- One user field could be used to accumulate an installation accounting unit. For example, you might count different amounts for different types of transaction. Or, in a browsing application, you might count 1 unit for each record scanned and not selected, and 3 for each record selected.

  You can also treat the fullword count fields as 32-bit flag fields to indicate special situations, for example, out-of-line situations in the applications, operator errors, and so on. CICS includes facilities to turn individual bits or groups of bits on or off in these counts.
- The performance clocks can be used for accumulating the time taken for I/O, DL/I scheduling, and so on. It usually includes any waiting for the transaction to regain control after the requested operation has completed. Because the periods are counted as well as added, you can get the average time waiting for I/O as well as the total. If you want to highlight an unusually long individual case, set a flag on in a user count as explained above.
- One use of the performance character string is for systems in which one transaction ID is used for widely differing functions. The application can enter a subsidiary ID into the string to indicate which particular variant of the transaction applies in each case.

  Some users have a single transaction ID so that all user input is routed through a common prologue program for security checking, for example. In this case, it is very easy to record the subtransaction identifier during this prologue. (However, it is equally possible to route transactions with different identifiers to the same program, in which case this technique is not necessary.)

Note: Accounting data for a single combination of either transaction, terminal, or operator identifiers may represent multiple transactions. In a system with dial-up or pooled TCAM terminals, or in an SNA network using pipeline sessions, one

operator may work with different terminal identifiers at different times, so each accounting record may have contributions from several operators.

The monitoring control table (MCT)

You use the monitoring control table (MCT):

- To tell CICS about the EMPs that you have coded in your application programs and about the data that is to be collected at these points
- To tell CICS that you want certain system-defined performance data not to be recorded during a particular CICS run.

DFHMCT TYPE=EMP

There must be a DFHMCT TYPE=EMP macro definition for every user-coded EMP. This macro has an ID operand, whose value must be made up of the ENTRYNAME and POINT values specified on the EXEC CICS MONITOR command. The PERFORM operand of the DFHMCT TYPE=EMP macro tells CICS which user count fields, user clocks, and character values to expect at the identified user EMP, and what operations to perform on them.

DFHMCT TYPE=RECORD

The DFHMCT TYPE=RECORD macro allows you to *exclude* specific system-defined performance data from a CICS run. (Each performance monitoring record is approximately 1288 bytes long, without taking into account any user data that may be added, or any excluded fields.)

Each field of the performance data that is gathered at the system-defined EMPs belongs to a group of fields that has a group identifier. Each performance data field also has its own numeric identifier that is unique within the group identifier. For example, the transaction sequence number field in a performance record belongs to the group DFHTASK, and has the numeric identifier '031'. Using these identifiers, you can exclude specific fields or groups of fields, and reduce the size of the performance records.

Full details of the MCT are provided in the *CICS/ESA Resource Definition Guide*, and examples of MCT coding are included with the programming information in the *CICS/ESA Customization Guide*.

Three sample monitoring control tables are also provided in CICS410.SDFHSAMP:

- For terminal-owning regions (TORs) - DFHMCTT$
- For application-owning regions (AORs) - DFHMCTA$
- For file-owning regions (FORs) - DFHMCTF$.

These samples show how to use the EXCLUDE and INCLUDE operands to reduce the size of the performance class record in order to reduce the volume of data written by CICS to SMF.

Controlling CICS monitoring

When CICS is initialized, you switch the monitoring facility on by specifying the system initialization parameter MN=ON. MN=OFF is the default setting. You can select the classes of monitoring data you want to be collected using the MNPER, MNEXC, and MNEVE system initialization parameters. You can request the collection of any combination of performance class data, exception class data, and SYSEVENT data. The class settings can be changed whether monitoring itself is ON or OFF. For guidance about system initialization parameters, refer to the *CICS/ESA System Definition Guide*.

When CICS is running, you can control the monitoring facility dynamically. Just as at CICS initialization, you can switch monitoring on or off, and you can change the classes of monitoring data that are being collected. There are two ways of doing this:

1. You can use the master terminal CEMT INQ|SET MONITOR command, which is described in the *CICS/ESA CICS-Supplied Transactions* manual.
2. You can use the EXEC CICS INQUIRE and SET MONITOR commands; programming information about these is in the *CICS/ESA System Programming Reference*.

If you activate a class of monitoring data in the middle of a run, the data for that class becomes available only for transactions that are started thereafter. You cannot change the classes of monitoring data collected for a transaction after it has started. It is often preferable, particularly for long-running transactions, to start all classes of monitoring data at CICS initialization.

Processing of CICS monitoring facility output

See Chapter 7, “Service Level Reporter (SLR)” on page 95 for more information.

Or, instead, you may want to write your own application program to process output from the CICS monitoring facility. The *CICS/ESA Customization Guide* gives programming information about the format of this output.

CICS provides a sample program, DFH$MOLS, which reads, formats, and prints monitoring data. It is intended as a sample program that you can use as a skeleton if you need to write your own program to analyze the data set. Comments within the program may help you if you want to do your own processing of CICS monitoring facility output. See the *CICS/ESA Operations and Utilities Guide* for further information on the DFH$MOLS program.

Performance implications

For information on the performance implications of using the CICS monitoring facility, see “Resource percentile for LSR (RSCLMT)” on page 227.

Interpreting CICS monitoring

All of the exception class data and all of the system-defined performance class data that can be produced by CICS monitoring is listed below. Each of the data fields is presented as a field description, followed by an explanation of the contents. The field description has the format shown in Figure 4, which is taken from the performance data group DFHTASK.

```
001 (TYPE-C, 'TRAN', 4 BYTES)
 |      |       |       |
 |      |       |      Length of the field (as re-
 |      |       |      presented by CMODLENG in the
 |      |       |      dictionary entry).
 |      |       |
 |      |      Informal name for the field, as used,
 |      |      perhaps, in column headings when the
 |      |      monitoring output is postprocessed
 |      |      (CMODHEAD of the dictionary entry).
 |      |
 |     Data type, which may be one of the following:
 |       A - a 32-bit count
 |       C - a byte string
 |       P - a packed decimal value
 |       S - a clock comprising a 32-bit accumulation
 |           of 16-microsecond units followed by an
 |           8-bit flag followed by a 24-bit count
 |           (modulo-16 777 216) of the number of
 |           intervals included in the accumulation.
 |       T - a time stamp derived directly from the
 |           output of an STCK instruction.
 |      (CMODTYPE of the dictionary entry)
 |
Field identifier by which the field may be individually
excluded or included during MCT preparation (CMODIDNT of
the dictionary entry).
```

Figure 4. Format of the descriptions of the data fields

Note: References in Figure 4 to the associated dictionary entries apply only to the performance class data descriptions. Exception class data is not defined in the dictionary record.

Clocks and time stamps

In the descriptions that follow, the term **clock** is distinguished from the term **time stamp**.

A **clock** is a 32-bit value, expressed in units of 16 microseconds, accumulated during one or more measurement periods. The 32-bit value is followed by 8 reserved bits, which are in turn followed by a 24-bit value indicating the number of such periods.

Neither the 32-bit timer component of a clock nor its 24-bit period count are protected against wraparound. The timer capacity is about 18 hours, and the period count runs modulo 16 777 216.

The 8 reserved bits have the following significance:

Bits 0, 1, 2 and 3	Used for online control of the clock when it is running, and should always be zeros on output.
Bits 4 and 7	Not used.
Bits 5 and 6	Used to indicate, when set to 1, that the clock has suffered at least one out-of-phase start (bit 5) or stop (bit 6).

A **time stamp** is an 8-byte copy of the output of an STCK instruction.

Note: All times produced in the offline reports are in GMT (Greenwich Mean Time) not local time. Times produced by online reporting can be expressed in either GMT or local time.

Performance class data

The performance class data is described below in order of group name. The group name is always in field CMODNAME of the dictionary entry.

A user task can be represented by one or more performance class monitoring records, depending on whether the MCT event monitoring option DELIVER or the system initialization parameters MNCONV=YES or MNSYNC=YES have been selected. In the descriptions that follow, the term “user task” means “that part or whole of a transaction that is represented by a performance class record”, unless the description states otherwise.

A note about response time

You can calculate the internal CICS response time by subtracting performance data field 005 (start time) from performance data field 006 (finish time).

Figure 5 shows the relationship of dispatch time, suspend time, and CPU time with the response time.

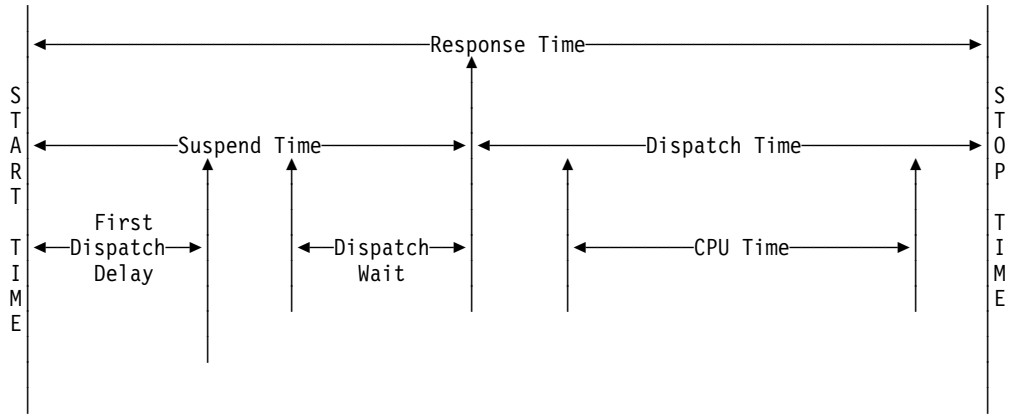


Figure 5. Response time relationships

A note about wait times

The performance data fields 009, 010, 011, 063, 100, 101, 129, 133, 134, 156, and 171. all record the elapsed time spent waiting for a particular type of I/O operation. For example, field 009 records the elapsed time waiting for terminal I/O. The elapsed time includes not only that time during which the I/O operation is actually taking place, but also the time during which the access method is completing the outstanding event control block, and the time subsequent to that until the waiting CICS transaction is redispatched.

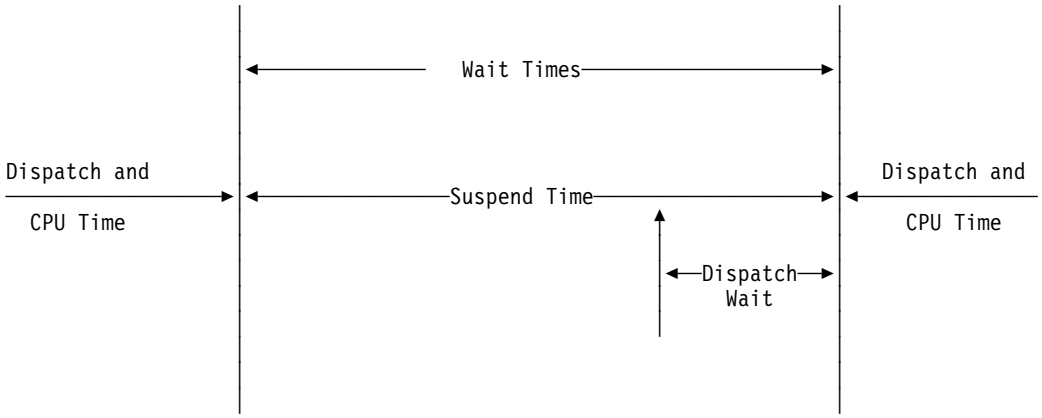


Figure 6. Wait time relationships

A note about program load time

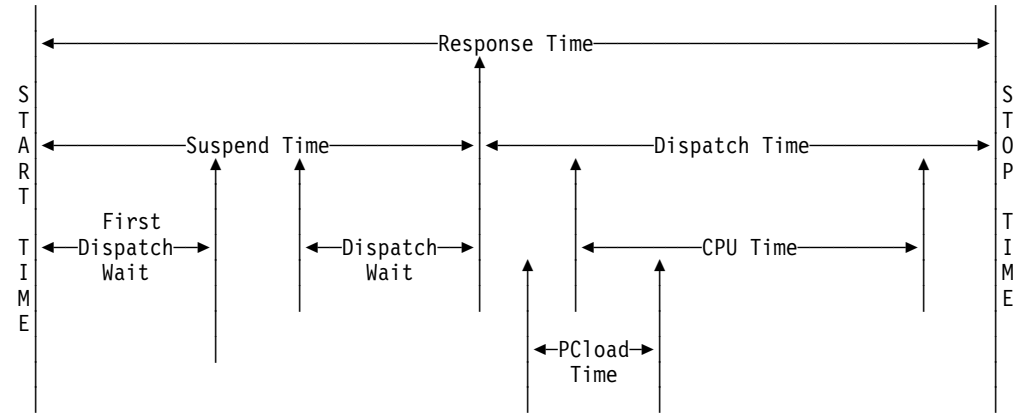


Figure 7. Program load time

Figure 7 shows the relationship between the program load time (field 115) and the dispatch time and the suspend time (fields 7 and 14).

A note about exception wait time

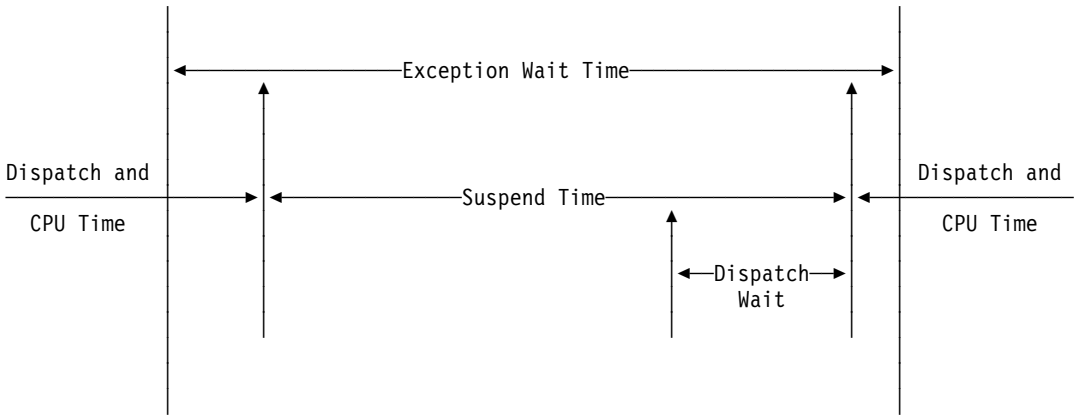


Figure 8. Exception wait time

Figure 8 shows the relationship between the exception wait time (field 103), and the dispatch time and suspend time (fields 7 and 14).

A note about RMI elapsed and suspend time

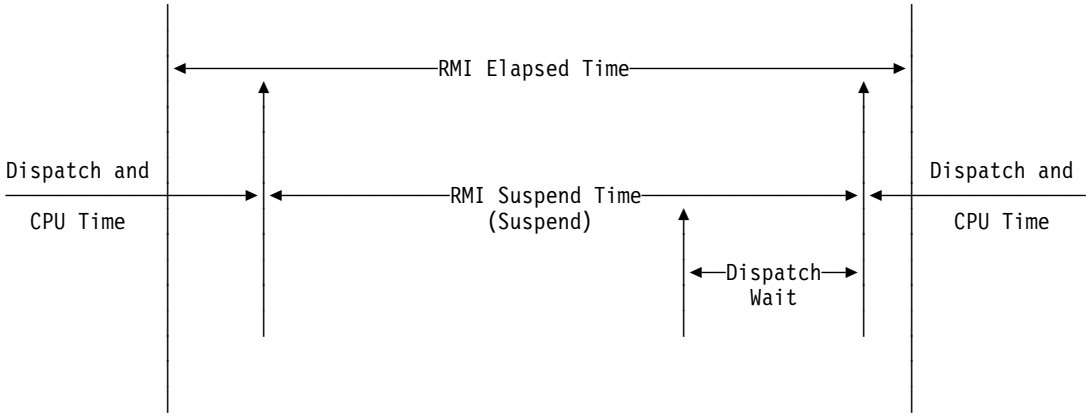


Figure 9. RMI elapsed and suspend time

Figure 9 shows the relationship between the RMI elapsed time and the suspend time (fields 170 and 171).

A note about storage occupancy counts

An occupancy count measures the area under the curve of user-task storage in use against elapsed time. The unit of measure is the “byte-unit”, where the “unit” is equal to 1024 microseconds, or 1.024 milliseconds. Where *ms* is milliseconds, a user task occupying, for example, 256 bytes for 125 milliseconds, is measured as follows:

125 / 1.024 *ms* = 122 units * 256 = 31 232 byte-units.

Note: All references to “Start time” and “Stop time” in the calculations below refer to the middle 4 bytes of each 8 byte start/stop time field. Bit 51 of Start time or Stop time represents a unit of 16 microseconds.

To calculate response time and convert into microsecond units:
Response = ((Stop time – Start time) * 16)
To calculate number of 1024 microsecond “units”:
Units = (Response / 1024)
or
Units = ((Stop time – Start time) / 64)
To calculate the average user-task storage used from the storage occupancy count:
Average user-task = storage used (Storage Occupancy / Units)

To calculate units per second:
Units Per Second = (1 000 000 / 1024) = 976.5625
To calculate the response time in seconds:
Response time = (((Stop time – Start time) * 16) / 1 000 000)

During the life of a user task CICS measures, calculates, and accumulates the storage occupancy at the following points:

- Before GETMAIN increases current user-storage values
- Before FREEMAIN reduces current user-storage values
- Just before the performance record is moved to the buffer.

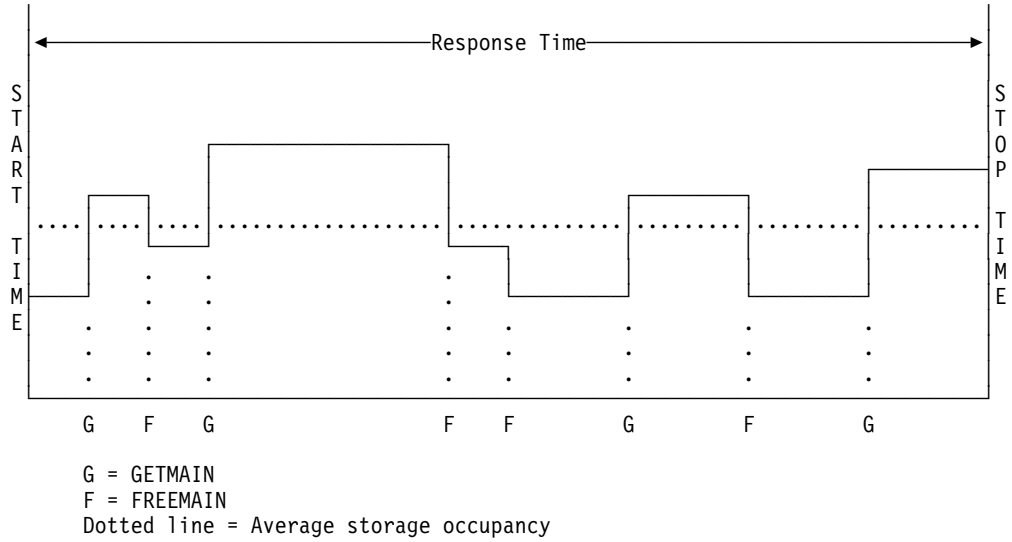


Figure 10. Storage occupancy

A note about program storage

The level of program storage currently in use is incremented at LOAD, LINK, and XCTL events by the size (in bytes) of the referenced program, and is decremented at RELEASE or RETURN events.

Note: On an XCTL event, the program storage currently in use is also decremented by the size of the program issuing the XCTL, because the program is no longer required.

Figure 11 shows the relationships between the "high-water mark" data fields that contain the maximum amounts of program storage in use by the user task. Field PCSTGHWM (field ID 087) contains the maximum amount of program storage in use by the task both above **and** below the 16MB line. Fields PC31AHWM (139) and PC24BHWM (108) are subsets of PCSTGHWM, containing the maximum amounts in use above and below the 16MB line, respectively. Further subset-fields contain the maximum amounts of storage in use by the task in each of the CICS dynamic storage areas (DSAs).

Note: The totaled values of all the subsets in a superset may not necessarily equate to the value of the superset; for example, the value of PC31AHWM plus the value of PC24BHWM may not equal the value of PCSTGHWM. This is because the peaks in the different types of program storage acquired by the user task do not necessarily occur simultaneously.

The "high-water mark" fields are described in detail in "Performance data in group DFHSTOR" on page 83.

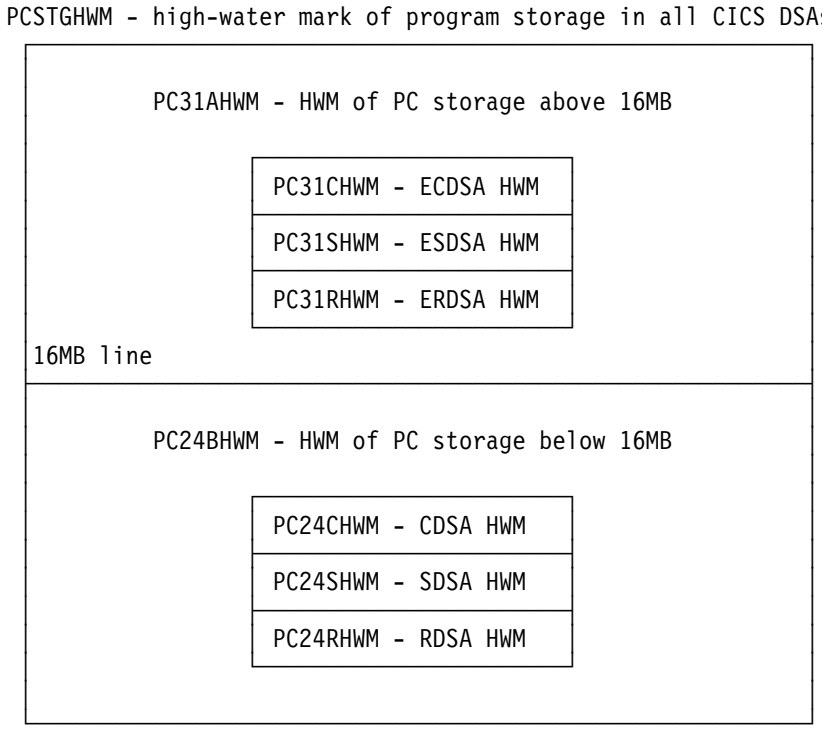


Figure 11. Relationships between the "high-water mark" program storage data fields

Performance data in group DFHCICS

005 (TYPE-T, 'START', 8 BYTES)
Start time of measurement interval. This is one of the following:

- The time at which the user task was attached
- The time at which data recording was most recently reset in support of the MCT user event monitoring point DELIVER option
- The monitoring options MNCONV, MNSYNC or FREQUENCY.

For more information see "Clocks and time stamps" on page 73.

006 (TYPE-T, 'STOP', 8 BYTES)
Finish time of measurement interval. This is either the time at which the user task was detached, or the time at which data recording was completed in support of the MCT user event monitoring point DELIVER option or the monitoring options MNCONV, MNSYNC or FREQUENCY. For more information see "Clocks and time stamps" on page 73.

Notes:

1. The measurement interval includes transaction dispatch time and suspend time values, if appropriate, and any first dispatch delay due to CMXT (if this is defined in a TCLASS) or MXT conditions.
2. Response Time = STOP – START.

For more information see "A note about response time" on page 74.

089 (TYPE-C, 'USERID', 8 BYTES)
User identification at task creation. This can also be the remote user identifier for a task created as the result of receiving an ATTACH request across an MRO or APPC link with attach-time security enabled.

103 (TYPE-S, 'EXWTTIME', 8 BYTES)
Accumulated data for exception conditions. The 32-bit clock contains the total elapsed time for which the user waited on exception conditions. The 24-bit period count equals the number of exception conditions that have occurred for this task. For more information see "Clocks and time stamps" on page 73.

Note: The performance class data field 'exception wait time' will be updated when exception conditions are encountered even when the exception class is inactive. For more information see Figure 8 on page 76.

112 (TYPE-C, 'RTYPE', 4 BYTES)
Performance record type (low-order byte-3):

C Record output for a terminal converse
D Record output for a user EMP DELIVER request
F Record output for a long-running transaction
S Record output for a syncpoint
T Record output for a task termination.

130 (TYPE-C, 'RSYSID', 4 bytes)
is the name (sysid) of the remote system to which this transaction was routed either statically or dynamically.

Note: If the transaction was not routed or was routed locally, this field is set to null. Also see the program name (field 71).

Performance data in group DFHDEST

041 (TYPE-A, 'TDGETCT', 4 BYTES)
Number of transient data GET requests issued by the user task.

042 (TYPE-A, 'TDPUTCT', 4 BYTES)
Number of transient data PUT requests issued by the user task.

043 (TYPE-A, 'TDPURCT', 4 BYTES)
Number of transient data PURGE requests issued by the user task.

091 (TYPE-A, 'TDTOTCT', 4 BYTES)
Total number of transient data requests issued by the user task. This field is the sum of TDGETCT, TDPUTCT, and TDPURCT.

101 (TYPE-S, 'TDIOWTT', 8 BYTES)
Elapsed time in which the user waited for VSAM transient data I/O. For more information see "Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

Performance data in group DFHFEPI

150 (TYPE-A,'SZALLOCT', 4 bytes)
Number of conversations allocated by the user task. This number is incremented for each FEPI ALLOCATE POOL or FEPI CONVERSE POOL.

151 (TYPE-A,'SZRCVCT', 4 bytes)
Number of FEPI RECEIVE requests made by the user task. This number is also incremented for each FEPI CONVERSE request.

152 (TYPE-A,'SZSENDCT', 4 bytes)
Number of FEPI SEND requests made by the user task. This number is also incremented for each FEPI CONVERSE request.

153 (TYPE-A,'SZSTRTCT', 4 bytes)
Number of FEPI START requests made by the user task.

154 (TYPE-A,'SZCHROUT', 4 bytes)
Number of characters sent through FEPI by the user task.

155 (TYPE-A,'SZCHRIN', 4 bytes)
Number of characters received through FEPI by the user task.

156 (TYPE-S,'SZWAIT', 8 bytes)
Elapsed time in which the user task waited for all FEPI services. For more information see "Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

157 (TYPE-A,'SZALLCTO', 4 bytes)
Number of times the user task timed out while waiting to allocate a conversation.

158 (TYPE-A,'SZRCVTO', 4 bytes)
Number of times the user task timed out while waiting to receive data.

159 (TYPE-A,'SZTOTCT', 4 bytes)
Total number of all FEPI API and SPI requests made by the user task.

Performance data in group DFHFILE

036 (TYPE-A, 'FCGETCT', 4 BYTES)
Number of file GET requests issued by the user task.

037 (TYPE-A, 'FCPUTCT', 4 BYTES)
Number of file PUT requests issued by the user task.

038 (TYPE-A, 'FCBRWCT', 4 BYTES)
Number of file browse requests issued by the user task. This number excludes the START and END browse requests.

039 (TYPE-A, 'FCADDCT', 4 BYTES)
Number of file ADD requests issued by the user task.

040 (TYPE-A, 'FCDELCT', 4 BYTES)
Number of file DELETE requests issued by the user task.

063 (TYPE-S, 'FCIOWTT', 8 BYTES)
Elapsed time in which the user task waited for file I/O. For more information see "Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

070 (TYPE-A, 'FCAMCT', 4 BYTES)
Number of times the user task invoked file access-method interfaces. This number excludes requests for OPEN and CLOSE.

093 (TYPE-A, 'FCTOTCT', 4 BYTES)
Total number of file control requests issued by the user task. This number excludes any request for OPEN, CLOSE, ENABLE, or DISABLE of a file.

How EXEC CICS file commands correspond to file control monitoring fields:

Table 5. EXEC CICS file commands related to file control monitoring fields

EXEC CICS command	Monitoring fields
READ	FCGETCT and FCTOTCT
READ UPDATE	FCGETCT and FCTOTCT
DELETE (after READ UPDATE)	FCDELCT and FCTOTCT
DELETE (with RIDFLD)	FCDELCT and FCTOTCT
REWRITE	FCPUTCT and FCTOTCT
WRITE	FCADDCT and FCTOTCT
STARTBR	FCTOTCT
READNEXT	FCBRWCT and FCTOTCT
READPREV	FCBRWCT and FCTOTCT
ENDBR	FCTOTCT
RESETBR	FCTOTCT
UNLOCK	FCTOTCT

Note: The number of STARTBR, ENDBR, RESETBR and UNLOCK file control requests can be calculated by subtracting the file request counts, FCGETCT, FCPUTCT, FCBRWCT, FCADDCT, and FCDELCT from the total file request count, FCTOTCT.

Performance data in group DFHJOUR

010 (TYPE-S, 'JCIOWTT', 8 BYTES)
Elapsed time for which the user task waited for journal I/O. For more information see "Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

058 (TYPE-A, 'JCPUWRCT', 4 BYTES)
Number of journal output requests during the user task.

Performance data in group DFHMAPP

050 (TYPE-A, 'BMSMAPCT', 4 BYTES)
Number of BMS MAP requests issued by the user task. This field corresponds to the number of RECEIVE MAP requests that did not incur a terminal I/O, and the number of RECEIVE MAP FROM requests.

051 (TYPE-A, 'BMSINCT', 4 BYTES)
Number of BMS IN requests issued by the user task. This field corresponds to the number of RECEIVE MAP requests that incurred a terminal I/O.

052 (TYPE-A, 'BMSOUTCT', 4 BYTES)
Number of BMS OUT requests issued by the user task. This field corresponds to the number of SEND MAP requests.

090 (TYPE-A, 'BMSTOTCT', 4 BYTES)
Total number of BMS requests issued by the user task. This field is the sum of BMSMAPCT, BMSINCT, and BMSOUTCT counts.

Performance data in group DFHPROG

055 (TYPE-A, 'PCLINKCT', 4 BYTES)
Number of program LINK requests issued by the user task, including the link to the first program of the user task.

056 (TYPE-A, 'PCXCTLCT', 4 BYTES)
Number of program XCTL requests issued by the user task.

057 (TYPE-A, 'PCLOADCT', 4 BYTES)
Number of program LOAD requests issued by the user task.

071 (TYPE-C, 'PGMNAME', 8 BYTES)
The name of the first program invoked at attach-time.

For a remote transaction:

- If this CICS' definition of the remote transaction does not specify a program name, this field contains blanks.
- If this CICS' definition of the remote transaction specifies a program name, this field contains the name of the specified program. (Note that this is not necessarily the program that is run on the remote system.)

For a dynamically-routed transaction, if the dynamic transaction routing program routes the transaction locally and specifies an alternate program name, this field contains the name of the alternate program.

For a dynamic program link (DPL) mirror transaction, this field contains the initial program name specified in the dynamic program LINK request. DPL

mirror transactions can be identified using byte 1 of the transaction flags TRANFLAG (164) field.

113 (TYPE-C, 'ABCODEO', 4 BYTES)
Original abend code.

114 (TYPE-C, 'ABCODEC', 4 BYTES)
Current abend code.

115 (TYPE-S, 'PCLOADTM', 8 BYTES)
Elapsed time in which the user task waited for program library (DFHRPL) fetches. Only fetches for programs with installed program definitions or autoinstalled as a result of application requests are included in this figure. However, installed programs residing in the LPA are not included (because they do not incur a physical fetch from a library). For more information about program load time see "Clocks and time stamps" on page 73, and Figure 7 on page 75.

Performance data in group DFHSTOR

User storage fields in group DFHSTOR:

033 (TYPE-A, 'SCUSRHWM', 4 BYTES)
Maximum amount (high-water mark) of user storage allocated to the user task below the 16MB line, in the user dynamic storage area (UDSA).

054 (TYPE-A, 'SCUGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task below the 16MB line, in the UDSA.

095 (TYPE-A, 'SCUSRSTG', 8 BYTES)
Storage occupancy of the user task below the 16MB line, in the UDSA. This measures the area under the curve of storage in use against elapsed time. For more information about storage occupancy, see "A note about storage occupancy counts" on page 77.

105 (TYPE-A, 'SCUGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task for storage above the 16MB line, in the extended user dynamic storage area (EUDSA).

106 (TYPE-A, 'SCUSRHWM', 4 BYTES)
Maximum amount (high-water mark) of user-storage allocated to the user task above the 16MB line, in the EUDSA.

107 (TYPE-A, 'SCUCRSTG', 8 BYTES)
Storage occupancy of the user task above the 16MB line, in the EUDSA. This measures the area under the curve of storage in use against elapsed time. For more information see "A note about storage occupancy counts" on page 77.

116 (TYPE-A, 'SC24CHWM', 4 BYTES)
Maximum amount (high-water mark) of user-storage allocated to the user task below the 16MB line, in the CICS dynamic storage area (CDSA).

117 (TYPE-A, 'SCCGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task for storage below the 16MB line, in the CDSA.

118 (TYPE-A, 'SC24COCC', 8 BYTES)
Storage occupancy of the user task below the 16MB line, in the CDSA. This measures the area under the curve of storage in use against elapsed time. For more information see "A note about storage occupancy counts" on page 77.

119 (TYPE-A, 'SC31CHWM', 4 BYTES)
Maximum amount (high-water mark) of user-storage allocated to the user task above the 16MB line, in the extended CICS dynamic storage area (ECDSA).

120 (TYPE-A, 'SCCGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task for storage above the 16MB line, in the ECDSA.

121 (TYPE-A, 'SC31COCC', 8 BYTES)
Storage occupancy of the user task above the 16MB line, in the ECDSA. This measures the area under the curve of storage in use against elapsed time. For more information see "A note about storage occupancy counts" on page 77.

Table 6. User storage field id cross reference

	UDSA	EUDSA	CDSA	ECDSA
Getmain count	054	105	117	120
High-water-mark	033	106	116	119
Occupancy	095	107	118	121

Program storage fields in group DFHSTOR:

087 (TYPE-A, 'PCSTGHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task both above **and** below the 16MB line.

108 (TYPE-A, 'PC24BHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line. This field is a subset of PCSTGHWM (field ID 087) that resides below the 16MB line.

122 (TYPE-A, 'PC31RHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line, in the extended read-only dynamic storage area (ERDSA). This field is a subset of PC31AHWM (field ID 139) that resides in the ERDSA.

139 (TYPE-A, 'PC31AHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line. This field is a subset of PCSTGHWM (field ID 087) that resides above the 16MB line.

142 (TYPE-A, 'PC31CHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line, in the extended CICS dynamic storage area (ECDSA). This field is a subset of PC31AHWM (139) that resides in the ECDSA.

143 (TYPE-A, 'PC24CHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the CICS dynamic storage area (CDSA). This field is a subset of PC24BHWM (108) that resides in the CDSA.

160 (TYPE-A, 'PC24SHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the shared dynamic storage area (SDSA). This field is a subset of PC24BHWM (108) that resides in the SDSA.

161 (TYPE-A, 'PC31SHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line, in the extended shared dynamic storage area (ESDSA). This field is a subset of PC31AHWM (139) that resides in the ESDSA.

162 (TYPE-A, 'PC24RHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the read-only dynamic storage area (RDSA). This field is a subset of PC24BHWM (108) that resides in the RDSA.

Performance data in group DFHSYNC

060 (TYPE-A, 'SPSYNCCT', 4 BYTES)
Number of SYNCPOINT requests issued during the user task.

Notes:

1. A SYNCPOINT is implicitly issued as part of the task-detach processing.
2. A SYNCPOINT is issued at PSB termination for DL/I and DBCTL.

Performance data in group DFHTASK

001 (TYPE-C, 'TRAN', 4 BYTES)
Transaction identification.

004 (TYPE-C, 'T', 4 BYTES)
Transaction start type. The high-order bytes (0 and 1) are set to:

'TO ' Attached from terminal input
'S ' Attached by automatic transaction initiation (ATI) without data
'SD' Attached by automatic transaction initiation (ATI) with data
'QD' Attached by transient data trigger level
'U ' Attached by user request
'TP' Attached from terminal TCTTE transaction ID
'SZ' Attached by Front End Programming Interface (FEPI).

007 (TYPE-S, 'USRDISPT', 8 BYTES)
Elapsed time for which the user task was dispatched. For more information see "Clocks and time stamps" on page 73.

008 (TYPE-S, 'USRCPUT', 8 BYTES)
Processor time for which the user task was dispatched on each CICS TCB (QR, RO, CO, SZ). For more information see "Clocks and time stamps" on page 73.

014 (TYPE-S, 'SUSPTIME', 8 BYTES)
Total elapsed wait time for which the user task was suspended by the dispatcher. This includes:

- The elapsed time waiting for the first dispatch. This also includes any delay incurred because of the limits set for this transaction's transaction class (if any) or by the system parameter MXT being reached.

- The task suspend (wait) time.
- The elapsed time waiting for redispatch after a suspended task has been resumed.

For more information, see “Clocks and time stamps” on page 73, and “A note about wait times” on page 75.

031 (TYPE-P, ‘TRANNUM’, 4 BYTES)
Transaction identification number.

Note: The transaction number field is normally a 4-byte packed decimal number. However, some CICS system tasks are identified by special character 'transaction numbers', as follows:

- ‘ III’ for system initialization task
- ‘ JBS’ or ‘ Jnn’ for journal control (where nn = the journal number from 01 - 99
- ‘ TCP’ for terminal control.

These special identifiers are placed in bytes 2 through 4. Byte 1 is a blank (X'40') before the terminal control TCP identifier, and a null value (X'00') before the others.

059 (TYPE-A, ‘ICPUINCT’, 4 BYTES)
Number of interval control START or INITIATE requests during the user task.

064 (TYPE-A, ‘TASKFLAG’, 4 BYTES)
Task error flags, a string of 31 bits used for signaling unusual conditions occurring during the user task:

Bit 0 Reserved

Bit 1 Detected an attempt either to start a user clock that was already running, or to stop one that was not running

Bits 2–31 Reserved.

097 (TYPE-C, ‘NETNAME’, 20 BYTES)
Fully qualified name by which the originating system is known to the VTAM network. This name is assigned at attach time using either the NETNAME derived from the TCT (when the task is attached to a local terminal), or the NETNAME passed as part of an ISC APPC or IRC attach header. At least three padding bytes (X'00') are present at the right end of the name.

If the originating terminal is VTAM across an ISC APPC or IRC link, then the NETNAME is the *networkid.LUname* If the terminal is non-VTAM, then the NETNAME is *networkid.generic_applid*.

All originating information passed as part of an ISC LUTYPE6.1 attach header has the same format as the non-VTAM terminal originators above.

When the originator is communicating over a DL/I batch session, the name is a concatenation of ‘jobname.stepname.procname’ derived from the originating system. Characters in excess of 17 cause truncation **at the left**.

When the originator is communicating over an external CICS interface (EXCI) session, the name is a concatenation of:

```
'DFHEXCIU | .       |  MVS Id    |  Address Space Id (ASID)'
 8 bytes  | 1 byte  |  4 bytes   |  4 bytes
```

derived from the originating system. That is, the name is a 17-byte LU name consisting of:

- An 8-byte eye-catcher set to 'DFHEXCIU'.
- A 1-byte field containing a period (.).
- A 4-byte field containing the MVSID, in characters, under which the client program is running.
- A 4-byte field containing the address space id (ASID) in which the client program is running. This field contains the 4-character EBCDIC representation of the 2-byte hex address space id.

098 (TYPE-C, 'UOWID', 8 BYTES)
Name by which the unit of work is known within the originating system. This name is assigned at attach time using either an STCK-derived token (when the task is attached to a local terminal), or the unit of work ID passed as part of an ISC APPC or IRC attach header.

The first six bytes of this field are one of the following:

- A binary value derived from the clock of the originating system and wrapping round at intervals of several months
- A character value of the form "hhmmss", which wraps round daily. This case applies when the originating system is communicating through a DL/I batch session.

The last two bytes of this field are for the period count. These may change during the life of the task as a result of syncpoint activity.

Note: When using MRO or ISC, the UOWID field must be combined with the NETNAME field (097) to uniquely identify a task, because UOWID is unique only to the originating CICS system.

102 (TYPE-S, 'DISPWTT', 8 BYTES)
Elapsed time for which the user task waited for redispatch. This is the aggregate of the wait times between each event completion and user-task redispatch.

Note: This field does not include the elapsed time spent waiting for first dispatch. This field is a subset of the task suspend time, SUSPTIME (014), field.

109 (TYPE-C, 'TRANPRI', 4 BYTES)
Transaction priority when monitoring of the task was initialized (low-order byte-3).

125 (TYPE-S, 'DSPDELAY', 8 BYTES)
The elapsed time waiting for first dispatch.

Note: This field is a subset of the task suspend time, SUSPTIME (014), field. For more information see "Clocks and time stamps" on page 73.

126 (TYPE-S, 'TCLDELAY', 8 BYTES)
The elapsed time waiting for first dispatch which was delayed because of the limits set for this transaction's transaction class, TCLSNAME (166), being reached. For more information see "Clocks and time stamps" on page 73.

Note: This field is a subset of the first dispatch delay, DSPDELAY (125), field.

127 (TYPE-S, 'MXTDELAY', 8 BYTES)
The elapsed time waiting for first dispatch which was delayed because of the limits set by the system parameter, MXT, being reached.

Note: The field is a subset of the first dispatch delay, DSPDELAY (125), field.

129 (TYPE-S, 'ENQDELAY', 8 BYTES)
The elapsed time waiting for a CICS Task Control ENQ. For more information see "Clocks and time stamps" on page 73.

Note: This field is a subset of the task suspend time, SUSPTIME (014), field.

166 (TYPE-C, 'TCLSNAME', 8 BYTES)
Transaction class name. This field is null if the transaction is not in a TRANCLASS.

170 (TYPE-S, 'RMITIME', 8 BYTES)
Amount of elapsed time spent in the Resource Manager Interface (RMI). For more information see "Clocks and time stamps" on page 73, and Figure 9 on page 76.

171 (TYPE-S, 'RMISUSP', 8 BYTES)
Amount of elapsed time the task was suspended by the dispatcher while in the Resource Manager Interface (RMI). For more information see "Clocks and time stamps" on page 73, "A note about wait times" on page 75, and Figure 9 on page 76.

Note: The field is a subset of the task suspend time, SUSPTIME (014), field and also the RMITIME (170) field.

Performance data in group DFHTERM

002 (TYPE-C, 'TERM', 4 BYTES)
Terminal or session identification. This field is null if the task is not associated with a terminal or session.

APAR

APAR PN71965 MJO 11/9/95 (4.1 only)

Note: If SURROGATE=YES has been coded on the DFHMCT TYPE=INITIAL statement, the term TERM field contains the surrogate terminal ID for a transaction routed task running in an AOR directly connected to a TOR.

009 (TYPE-S, 'TCIOWTT', 8 BYTES)
Elapsed time for which the user task waited for input from the terminal operator, after issuing a RECEIVE request. For more information see "Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

034 (TYPE-A, 'TCMSGIN1', 4 BYTES)
Number of messages received from the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

035 (TYPE-A, 'TCMSGOU1', 4 BYTES)
Number of messages sent to the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

067 (TYPE-A, 'TCMSGIN2', 4 BYTES)
Number of messages received from the LUTYPE6.1 alternate terminal facilities by the user task.

068 (TYPE-A, 'TCMSGOU2', 4 BYTES)
Number of messages sent to the LUTYPE6.1 alternate terminal facilities by the user task.

069 (TYPE-A, 'TCALLOCT', 4 BYTES)
Number of TCTTE ALLOCATE requests issued by the user task for LUTYPE6.2 (APPC), LUTYPE6.1, and IRC sessions.

083 (TYPE-A, 'TCCHRIN1', 4 BYTES)
| Number of characters received from the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

084 (TYPE-A, 'TCCHROU1', 4 BYTES)
| Number of characters sent to the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

085 (TYPE-A, 'TCCHRIN2', 4 BYTES)
Number of characters received from the LUTYPE6.1 alternate terminal facilities by the user task. *(Not applicable to ISC APPC.)*

086 (TYPE-A, 'TCCHROU2', 4 BYTES)
Number of characters sent to the LUTYPE6.1 alternate terminal facilities by the user task. *(Not applicable to ISC APPC.)*

100 (TYPE-S, 'IRIOWTT', 8 BYTES)
Elapsed time for which the user task waited for control at this end of an MRO link. For more information see "Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

111 (TYPE-C, 'LUNAME', 8 BYTES)
VTAM logical unit name (if available) of the terminal associated with this transaction. If the task is executing in an application-owning or file-owning region then the luname is the generic applid of the originating connection for LUTYPE6.1 and LUTYPE6.2 (APPC). The luname is blank if the originating connection is either MRO (ISC) or external CICS interface (EXCI).

APAR

APAR PN71965 MJO 11/9/95 (4.1 only)

Note: If SURROGATE=YES has been coded on the DFHMCT TYPE=INITIAL statement, and this monitoring record is for a transaction routed task running in an AOR directly connected to a TOR, the LUNAME field will not relate to the surrogate terminal ID recorded in the TERM monitoring field. It relates to the session ID.

133 (TYPE-S, 'LU61WTT', 8 BYTES)
| The elapsed time for which the user task waited for I/O on a LUTYPE6.1 connection or session. This time also includes the waits incurred for conversations across LUTYPE6.1 connections, but not the waits incurred due to LUTYPE6.1 syncpoint flows. For more information see "Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

134 (TYPE-S, 'LU62WTT', 8 BYTES)
| The elapsed time for which the user task waited for I/O on a LUTYPE6.2 (APPC) connection or session. This time also includes the waits incurred for conversations across LUTYPE6.2 (APPC) connections, but not the waits incurred due to LUTYPE6.2 (APPC) syncpoint flows. For more information see

"Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

135 (TYPE-A, 'TCM62IN2', 4 BYTES)
Number of messages received from the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

136 (TYPE-A, 'TCM62OU2', 4 BYTES)
Number of messages sent to the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

137 (TYPE-A, 'TCC62IN2', 4 BYTES)
Number of characters received from the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

138 (TYPE-A, 'TCC62OU2', 4 BYTES)
Number of characters sent to the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

Performance data in group DFHTEMP

011 (TYPE-S, 'TSIOWTT', 8 BYTES)
Elapsed time for which the user task waited for VSAM temporary storage I/O. For more information see "Clocks and time stamps" on page 73, and "A note about wait times" on page 75.

044 (TYPE-A, 'TSGETCT', 4 BYTES)
Number of temporary-storage GET requests issued by the user task.

046 (TYPE-A, 'TSPUTACT', 4 BYTES)
Number of PUT requests to auxiliary temporary storage issued by the user task.

047 (TYPE-A, 'TSPUTMCT', 4 BYTES)
Number of PUT requests to main temporary storage issued by the user task.

092 (TYPE-A, 'TSTOTCT', 4 BYTES)
Total number of temporary-storage requests issued by the user task. This field is the sum of TSGETCT, TSPUTACT, and TSPUTMCT.

Exception class data

Exception records are produced after each of the following conditions encountered by a transaction has been resolved:

- Wait for storage in the CDSA
- Wait for storage in the UDSA
- Wait for storage in the SDSA
- Wait for storage in the RDSA
- Wait for storage in the ECDSA
- Wait for storage in the EUDSA
- Wait for storage in the ESDSA
- Wait for storage in the ERDSA
- Wait for auxiliary temporary storage
- Wait for auxiliary temporary storage string
- Wait for auxiliary temporary storage buffer
- Wait for file string
- Wait for file buffer
- Wait for LSRPOOL string.

These records are fixed format. The format is as follows:

```
MNEXCDS  DSECT
EXCMNTRN DS    CL4               TRANSACTION IDENTIFICATION
EXCMNTER DS    XL4               TERMINAL IDENTIFICATION
EXCMNUSR DS    CL8               USER IDENTIFICATION
EXCMNTST DS    CL4               TRANSACTION START TYPE
EXCMNSTA DS    XL8               EXCEPTION START TIME
EXCMNSTO DS    XL8               EXCEPTION STOP TIME
EXCMNTNO DS    PL4               TRANSACTION NUMBER
EXCMNTPR DS    XL4               TRANSACTION PRIORITY
         DS    CL4               RESERVED
EXCMNLUN DS    CL8               LUNAME
         DS    CL4               RESERVED
EXCMNEXN DS    XL4               EXCEPTION NUMBER
EXCMNRTY DS    CL8               EXCEPTION RESOURCE TYPE
EXCMNRID DS    CL8               EXCEPTION RESOURCE ID
EXCMNTYP DS    XL2               EXCEPTION TYPE
EXCMNWT  EQU   X'0001'             WAIT
EXCMNBWT EQU   X'0002'             BUFFER WAIT
EXCMNSWT EQU   X'0003'             STRING WAIT
         DS    CL2               RESERVED
EXCMNTCN DS    CL8               TRANSACTION CLASS NAME
```

Exception data

EXCMNTRN (TYPE-C, 4 BYTES)
Transaction identification. This field is null if the task is not associated with a terminal or session.

EXCMNTER (TYPE-C, 4 BYTES)
Terminal identification. This field is null if the task is not associated with a terminal or session.

EXCMNUSR (TYPE-C, 8 BYTES)
User identification at task creation. This can also be the remote user identifier for a task created as the result of receiving an ATTACH request across an MRO or APPC link with attach-time security enabled.

EXCMNTST (TYPE-C, 4 BYTES)
Transaction start type. The high-order byte (0 and 1) is set to:

'TO' Attached from terminal input
'S' Attached by automatic transaction initiation (ATI) without data
'SD' Attached by automatic transaction initiation (ATI) with data
'QD' Attached by transient data trigger level
'U ' Attached by user request
'TP' Attached from terminal TCTTE transaction ID
'SZ' Attached by Front End Programming Interface (FEPI).

EXCMNSTA (TYPE-T, 8 BYTES)
Start time of the exception.

EXCMNSTO (TYPE-T, 8 BYTES)
Finish time of the exception.

Note: The performance class exception wait time field, EXWTTIME (103), is a calculation based on subtracting the start time of the exception (EXCMNSTA) from the finish time of the exception (EXCMNSTO).

EXCMNTNO (TYPE-P, 4 BYTES)
Transaction identification number.

EXCMNTPR (TYPE-C, 4 BYTES)
Transaction priority when monitoring was initialized for the task (low-order byte).

EXCMNLUN (TYPE-C, 4 BYTES)
VTAM logical unit name (if available) of the terminal associated with this transaction. This field is nulls if the task is not associated with a terminal.

EXCMNEXN (TYPE-A, 4 BYTES)
Exception sequence number for this task.

EXCMNRTY (TYPE-C, 8 BYTES)
Exception resource type. The possible values for EXCMNRTY are shown in Table 7 on page 93.

EXCMNRID (TYPE-C, 8 BYTES)
Exception resource identification. The possible values for EXCMNRID are shown in Table 7 on page 93.

EXCMNTYP (TYPE-A, 2 BYTES)
Exception type. This field can be set to one of the following values:

X'0001' Exception due to a wait (EXCMNWT)
X'0002' Exception due to a buffer wait (EXCMNBWT)
X'0003' Exception due to a string wait (EXCMNSWT)

EXCMNTCN (TYPE-C, 8 BYTES)
Transaction class name. This field is null if the transaction is not in a transaction class.

The following table shows the value and relationships of the fields EXCMNTYP, EXCMNRTY, and EXCMNRID.

Table 7. Possible values of EXCMNTYP, EXCMNRTY, and EXCMNRID. The relationship between exception type, resource type, and resource identification.

EXCMNTYP Exception type	**EXCMNRTY Resource type**	**EXCMNRID Resource ID**	**.br Meaning**
EXCMNWT	'STORAGE'	'UDSA'	Wait for UDSA storage
EXCMNWT	'STORAGE'	'EUDSA'	Wait for EUDSA storage
EXCMNWT	'STORAGE'	'CDSA'	Wait for CDSA storage
EXCMNWT	'STORAGE'	'ECDSA'	Wait for ECDSA storage
EXCMNWT	'STORAGE'	'SDSA'	Wait for SDSA storage
EXCMNWT	'STORAGE'	'ESDSA'	Wait for ESDSA storage
EXCMNWT	'STORAGE'	'RDSA'	Wait for RDSA storage
EXCMNWT	'STORAGE'	'ERDSA'	Wait for ERDSA storage
EXCMNWT	'TEMPSTOR'	TS Qname	Wait for temporary storage
EXCMNSWT	'FILE'	filename	Wait for string associated with file
EXCMNSWT	'LSRPOOL'	filename	Wait for string associated with LSRPOOL
EXCMNSWT	'TEMPSTOR"	TS Qname	Wait for string associated with DFHTEMP
EXCMNBWT	'LSRPOOL'	LSRPOOL	Wait for buffer associated with LSRPOOL
EXCMNBWT	'TEMPSTOR'	TS Qname	Wait for buffer associated with DFHTEMP

End of Product-Sensitive programming interface

Chapter 7. Service Level Reporter (SLR)

Service Level Reporter (SLR), collects and analyzes data from CICS and other IBM products. You can summarize the data to any level and use many combinations of keys to select and sort records. You can build the reports online or in batch, and they can be in tabular or graphic form. A graphical presentation is especially useful for performance management, as it is much easier to see peaks, to spot correlations between data, and to see deviations from a straight line. SLR uses GDDM to produce its graphic reports.

You have complete flexibility when you build your own reports. However, SLR comes with a full set of reports to satisfy the following needs:

- System overviews
- Service levels
- Availability
- Performance
- Capacity planning
- Change and problem management
- Accounting.

SLR and CICS

SLR collects CICS performance and exception monitoring records and CICS statistics records (interval and end-of-day). SLR also collects the records written by RMF when CICS SYSEVENT recording is active. SLR analyzes and summarizes the data and stores the results in the SLR database.

You can look at detailed data, for example, all transactions entered from a selected terminal during a few minutes of the day, and data summarized for longer periods, for example hours, days, weeks, or months. It is possible to delete the old detailed data and keep only the higher-level summaries. The same SLR tables can therefore be used both for solving performance problems and for capacity planning.

SLR collects CICS data in the following ways:

- SLR automatically maintains multiple CICS dictionaries on its database, so records from different CICS releases can be processed in the same collect run.
- Data from multiple tasks can be combined into a unit of work. This means that you can link an original transaction with its subsidiary transactions started in another CICS system or region.

Normally, SLR reads the log records from a copy of the SMF data set, summarizes the data, and stores only the summarized version, to conserve disk space. If you need the detailed data for a special report, SLR can read the SMF log and produce reports directly from the input data.

There are a number of predefined reports, available both online (accessed through an ISPF dialog) and through batch jobs. You can also build your own reports, using either SLR.'s report generator or the SLR command language.

As an example, an SLR table called CICSTRANSUM contains the data from CICS performance monitoring records. Figure 12 shows part of a matrix report produced

from CICSTRANSUM. SLR can also display the same information graphically, using GDDM/PGF.

```
TRANSACTIONS │               MONTH
             │       JAN       FEB       MAR
*********************************************
ABCD         │       245      1700      2034              YEAR = 89
CECI         │     12003     14378     15432              MONTH ¬ = TOT
CEMT         │      2695      4204      5881              DAY = TOT
CSSF         │     15645     17654     15771              HOUR = TOT
CSSN         │     16545     17854     15802              APPLID = CICSTST
DE45         │     23775     32654     24190              TRANS ¬ = TOT
G321         │      3445      3210      2478              TERM = TOT
G322         │       451       566       648
```

Figure 12. Numbers of CICS transactions

SLR. supports the DL/I and DBCTL extensions to the performance monitoring records.

Application areas

You can use SLR to produce the following types of report:

One-page management reports: You can use the SLR-produced data to produce a series of reports that summarize CICS system behavior, giving processor and wait times and virtual storage usage. Such reports use data from many different types of log record.

Service levels: You can use SLR to check that the objectives in your service agreements are being met.

Availability: SLR can use log records to determine whether a resource is available. The resource can be a group of CICS systems (as with an MRO complex), a CICS system, or a resource within CICS such as a transaction. You specify the time that must elapse between log records before SLR considers a resource unavailable. You can use availability as part of your service level agreement with your users.

Performance: You can use the SLR-produced data to monitor response times, find peaks, and identify problems. SLR can produce:

- Detailed reports that give hour-by-hour task statistics on dispatch times, DL/I and I/O activity, wait times for individual CICS components, terminal activity, and other resources
- Summary reports that show day-by-day or month-by-month trends; for example, transaction statistics on response times, processor times, and character counts
- Exception reports that are available by time and by transaction code, as well as by terminal.

Capacity planning: SLR's ability to summarize data automatically means that you can keep months or even years of data in the database without taking up too much disk space. This gives you the opportunity to examine long term trends, which is useful for capacity planning. SLR can make predictions based on historical data.

Change and problem management: You can set up procedures for managing problems using the information handling capability of Information Management and the summarizing capability of SLR. For example, SLR can produce trend reports showing the relationship between the number of problems and rate of change activity.

Accounting: You can use SLR to charge users, using either a fixed price per transaction or a cost depending on the actual resources consumed. Accounting is based on performance monitoring records. SLR's accounting dialog helps you determine what prices you should set on CICS and other resources to cover your costs, based on unit price and pricing period.

Unload to IXF data format

You can unload CICS data collected by SLR to a file in the IXF data format. This enables processing by those products supporting the Integration Exchange Format (IXF):

- Application System (AS)
- Data Extract (DXT)
- Query Management Facility (QMF).

So you can use SLR to summarize log data and transfer the summarized data to DB2 database tables.

Sample CICS reports

SLR provides a number of reports that you can run immediately after you have installed SLR and collected data. The reports are similar to those produced for earlier CICS releases by the CICS/MVS Performance Analysis Reporting System for MVS (CICSPARS/MVS). Some of the reports are available online; others are available in batch. The reports are in graphical and numeric (tabular) format.

The reports use data from the following sources:

- CICS performance monitoring records
- CICS exception monitoring records
- RMF virtual storage data
- CICS statistics.

You run a supplied batch job to process the SMF log containing the above data. You normally run this job each night. It does not matter if the log contains data from several CICS systems, even though they may be different releases of CICS. SLR can also handle the cases where CICS is still active when the SMF log is dumped.

After collecting the data, the following reports are available:

Virtual storage reports: Virtual storage has always been a constraint for CICS systems. The provision of EDSAs above the 16MB line, the changes to program loading and compression, and the removal of the OSCOR parameter mean that it is especially important to monitor your use of virtual storage in the CICS address spaces.

These reports use RMF data and CICS statistics:

- Transaction storage
- Dynamic storage area
- Extended dynamic storage area
- Private storage, including LSQA, by subpool.

Performance monitoring reports: You can select most of these reports for particular transactions or terminals. This is useful where certain transactions are not giving the required performance, or a group of users are not getting the agreed service.

Find the cause of the problem by looking at the wait times for the various CICS components. You may find, for example, that file I/O is taking far too long, and further examination may point to the need for data set reorganization or moving a data set to another volume.

Where service times are not the cause of the problem, perhaps certain transactions do an excessive number of data transfers. The cause of this might be poor transaction design, or simply misuse.

Even if the sample reports do not show the fields you need, you can in seconds produce a tailored report showing exactly the fields you want, for the time period you are interested in.

These sample reports use CICS performance monitoring data and CICS statistics:

- Transaction response time
- Terminal control I/O count and wait time
- Temporary storage I/O count and wait time
- File control I/O count and wait time
- Transaction suspend count and suspend time
- Transaction rate
- Journal control I/O count and wait time
- Journal control requests and I/O counts
- Transient data requests
- Processor utilization
- File control access method calls
- Program control requests
- File control access method calls
- File control requests
- File control requests and I/O count
- BMS requests
- Transient data requests
- Terminal control requests and I/O count
- Temporary storage requests and I/O count
- Terminal control message count and message size.

Cross-system work report: This report uses data from CICS systems which communicate with each other using multiregion operation (MRO), combining data from tasks in the same unit of work.

Exception reports: Check these reports daily; they tell you about long transaction waits caused by a shortage of strings or buffers. You can usually avoid these bottlenecks by changing CICS parameters.

These reports (list and summary) are based on exception monitoring records.

Tailoring SLR

You can use the reporting commands to modify SLR's starter set reports and create your own reports. For example, you can:

- Print overviews, top lists, matrices, distributions, or graphic reports
- Change tabular formats to graphics and matrices, and vice versa
- Change selection criteria to get more or less detailed summaries
- Add explanatory text into a report
- Modify format defaults to your installation requirements
- Route the report output to appropriate locations.

You can add your own reports into the starter set of reports or replace the entire SLR dialog with your own.

By defining views of existing tables, you can:

- Change a table name or table description to be shown in reports
- Change the column names, headings, lengths, and formats
- Compute new columns from other columns in the same report
- Compute percentages based on totals
- Combine columns from several tables into overview reports
- Correlate data from several sources.

Editing commands allow you to:

- Display, insert, change, or delete data from your database
- Correct erroneous detail data and recalculate summaries
- Copy data from one table to another
- Create forecasts, and fit curves and trends.

The main purpose of modifying table descriptions is to change collect processing, for example, to process additional data not processed by the starter set.

Examples of what you can do are listed below:

- Modify your report headers and printing formats by changing:
  - A table name or description
  - A column name, description, edit, or unit specification.
- Report on additional data by:
  - Defining new columns computed from other columns in the same table
  - Adding a new data column to a summary or log table.
- Summarize data at extra levels by adding key columns.
- Summarize data over different time periods in a summary table by:
  - Adding or deleting time keys
  - Using weekly rather than monthly periods
  - Implementing shifts, or any user-defined time periods.
- Introduce new parameters by adding a parameter table to:
  - Define objectives for exception reporting
  - Define accounting prices that vary by shift or type of service
  - Translate codes into meaningful names
  - Group data by project or department names using accounting codes.
- Report on new data by adding new log tables and summary tables.

You can use the UNLOAD command to produce a sequential file of your database or selected tables from your database. You can specify which specific columns and rows to unload thus producing a desired subset of the SLR database for input to other programs. LOAD allows you to read unloaded tables back into your database.

Chapter 8. Enterprise Performance Data Manager/MVS

IBM SystemView Enterprise Performance Data Manager/MVS (EPDM) is a reporting system which uses DB2. It processes utilization and throughput statistics written to log data sets by computer systems, analyzes and stores the data into DB2, and presents it in a variety of forms. EPDM consists of a base product and several optional features that are used in systems management, as shown in Figure 13.

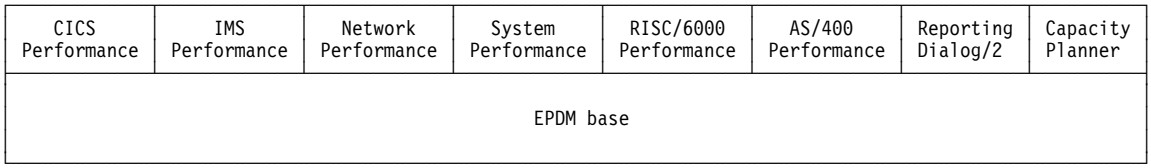


Figure 13. EPDM and optional features

The EPDM base includes:

- Reporting and administration dialogs that use the Interactive System Productivity Facility (ISPF)
- A collector function to read log data, with its own language
- Record mapping (definitions) for all data records used by the features

Each feature provides:

- Instructions (in the collector language) to transfer log data to DATABASE 2 (DB2) tables
- DB2 table definitions
- Reports

The EPDM database can contain data from many sources. For example, data from System Management Facilities (SMF), Resource Measurement Facility (RMF), CICS, and Information Management System (IMS) can be consolidated into a single report. In fact, you can define any non-standard log data to EPDM, and report on that data together with data coming from the standard sources.

The EPDM CICS Performance feature provides reports for your use when analyzing the performance of CICS/ESA and CICS/MVS, based on data from the CICS monitoring facility (CMF) and, for CICS/ESA, CICS statistics. These are some of the areas that EPDM can report on:

- Response times
- Resource usage
- Processor usage
- Storage usage
- Volumes and throughput
- CICS/DB2 activity
- Exceptions and incidents
- Data from connected regions, using the unit of work as key
- SYSEVENT data
- CICS availability
- CICS resource availability

The EPDM CICS Performance feature collects only the data required to meet CICS users' needs. You can combine that data with more data (called *environment data*), and present it in a variety of reports. EPDM provides an administration dialog for maintaining environment data. Figure 14 on page 102 illustrates how data is organized for presentation in EPDM reports.

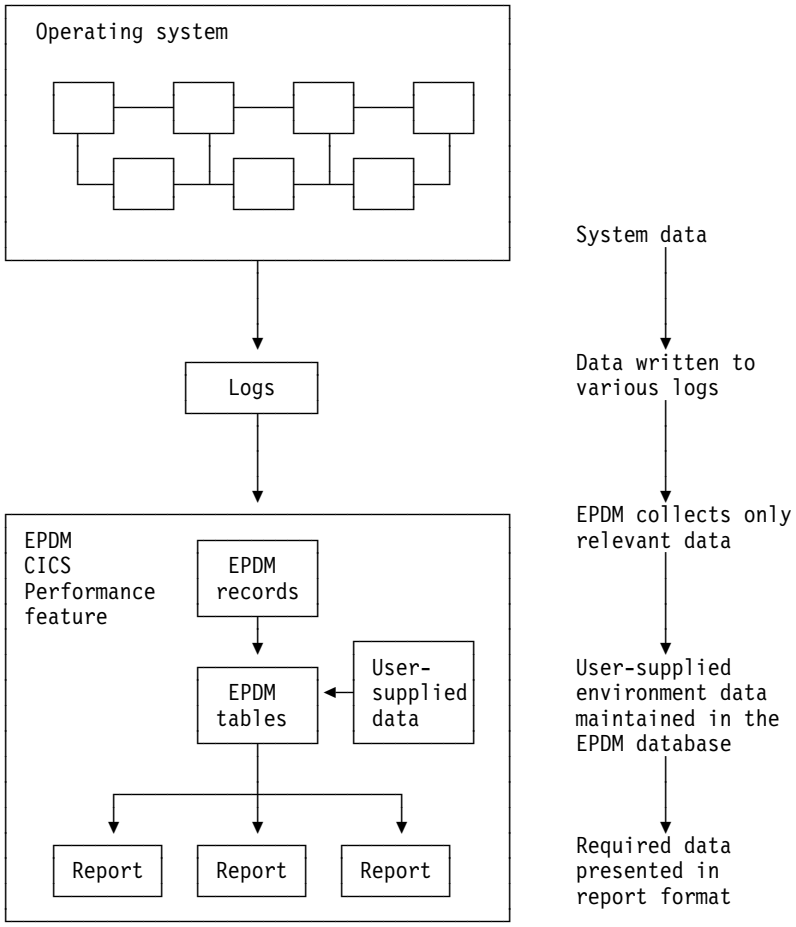


Figure 14. Organizing and presenting system performance data

The EPDM CICS Performance feature processes these records:

CMF

CICS/ESA performance
CICS/ESA exceptions
CICS/MVS accounting, performance, and exceptions

Statistics

CICS/ESA statistics

Using EPDM to report on CICS performance

To understand performance data, you must first understand the work CICS performs at your installation. Analyze the work by its basic building blocks: transactions. Group the transactions into categories of similar resource or user requirements and describe each category's characteristics. Understand the work that CICS performs for each transaction and the volume of transactions expected

during any given period. EPDM can show you various types of data for the transactions processed by CICS.

A service-level agreement for a CICS user group defines commitments in several areas of quantifiable CICS-related resources and services. CICS service commitments can belong to one of these areas:

- Response times
- Transaction rates
- Exceptions and incidents
- Availability

The following sections describe certain issues and concerns associated with systems management and how you can use the EPDM CICS Performance feature.

Monitoring response time

Use the EPDM CICS response-time reports to see the CICS application internal response times, whose elements are shown in Figure 15.

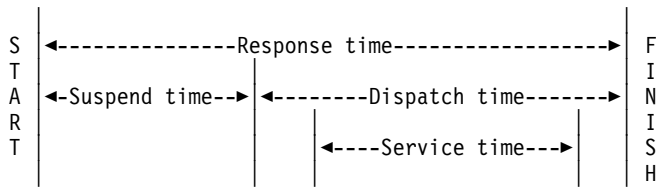


Figure 15. CICS internal response-time elements

As described in *EPDM Network Performance Feature Reports*, the Network Performance feature generates reports that show the total, end-to-end average response time (operator transit time) for VTAM applications (for example, a CICS region) by logical unit. The operator transit time consists of the host transit time and the network transit time, which are also shown in the Network Performance feature reports. Using these reports, you can isolate a response-time problem either to the network or to CICS and act on it accordingly. Should the problem be in CICS, you can use EPDM CICS Performance feature reports to identify the application causing the response-time degradation.

Monitoring processor and storage use

Poor response time usually indicates inefficient use of either the processor or storage (or both). EPDM-supplied reports can help you isolate a resource as the cause of a CICS performance problem.

If both the EPDM CICS Performance feature's statistics component and the EPDM System Performance feature's MVS component are installed and active, these reports are available for analyzing transaction rates and processor use by CICS region:

- The CICS Transaction Processor Utilization, Monthly report shows monthly averages for the dates you specify.
- The CICS Transaction Processor Utilization, Daily report shows daily averages for the dates you specify.

EPDM produces several reports that can help analyze storage usage. For example, the CICS Dynamic Storage (DSA) Usage report, shows pagepool usage.

```
                    CICS Dynamic Storage (DSA) Usage
                  MVS ID ='IPO2'      CICS ID ='CSRT5'
                   Date:  '1993-05-01' to '1993-05-02'


                                    Free     Free      Largest
Pagepool      DSA     Cushion     storage  storage      free
  name      (bytes)   (bytes)     (bytes)   (pct)       area    Getmains  Freemains
--------  ---------  --------  ---------  -------  ---------  --------  ---------
CDSA        1048576     65536     802816       76     765952      3695       3620
ECDSA       8388608    262144    7667712       91    7667712      8946       7252
ERDSA       3145728    262144    1302528       41    1290240       204          3
EUDSA       8388608    262144    8388608      100    8388608         1          1
UDSA        4194304     65536    4186112       99    4182016         6          4

                              EPDM Report: CICS809
```

Figure 16. CICS Dynamic storage (DSA) usage report

Reports such as this have the advantage that they do not need CMF data. Statistics incur less CICS processing overhead.

Monitoring volumes and throughput

Because CICS/ESA uses an MVS subtask to page and because an MVS page-in causes an MVS task to halt execution, the number of page-ins is a performance concern. Page-outs are not a concern because page-outs are scheduled to occur during lulls in CICS processing. If you suspect that a performance problem is related to excessive paging, you can use EPDM to report on page-ins, using RMF data.

The best indicator of a transaction's performance is its response. For each transaction ID, the CICS Transaction Performance, Detail report (in Figure 17) shows the total transaction count and the average response time.

```
                    CICS Transaction Performance, Detail
                     MVS ID ='IPO2'  CICS ID ='CFGTV1 '
                    Date: '1993-03-19'  to  '1993-03-20'


                  Avg      Avg   Prog                          Program
                  resp     CPU   load  Prog   FC               storage    Getmains  Getmains
Tran   Tran      time     time   reqs  loads calls  Excep-  bytes < 16 MB > 16 MB
 ID    count     (sec)    (sec) (avg) (avg) (avg)  tions    (max)       (avg)     (avg)
------------ -------- -------  ----  ----- -----  ------ ---------    --------  --------
QUIT    7916    0.085   0.017     0     0    18       0     74344          22         0
CRTE    1760    4.847   0.004     0     0     0       0    210176           1         0
AP00    1750    0.184   0.036     0     0     8       0    309800          66         0
PM94    1369    0.086   0.012     0     0     6       0    130096          24         0
VCS1     737    0.073   0.008     2     0     7       0     81200          14         0
PM80     666    1.053   0.155     1     0    62       0    104568         583         0
CESN     618    8.800   0.001     0     0     0       0     41608           0         0
SU01     487    0.441   0.062     4     0   126       0    177536          38         0
...
GC11       1    0.341   0.014     1     0     2       0     37048          10         0
DM08       1    0.028   0.002     0     0     0       0      5040           3         0
    ========                                                =========
       20359                                                309800

                              EPDM Report: CICS101
```

Figure 17. CICS transaction performance, detail report

Use this report to start verifying that you are meeting service-level objectives. First, verify that the values for average response time are acceptable. Then check that the transaction rates do not exceed agreed-to limits. If a transaction is not receiving the appropriate level of service, you must determine the cause of the delay.

Combining CICS and DB2 performance data

For each CICS task, CICS generates an LU6.2 unit-of-work ID. DB2 also creates an LU6.2 unit-of-work ID. Figure 18 shows how DB2 data can be correlated with CICS performance data using the DB2 token (QWHCTOKN) to identify the task.

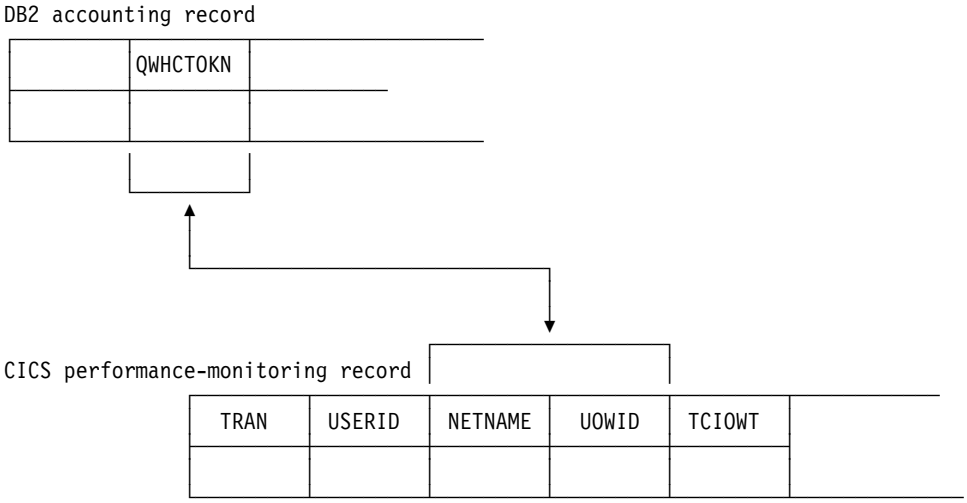


Figure 18. Correlating a CICS performance-monitoring record with a DB2 accounting record

If you match the NETNAME and UOWID fields in a CICS record to the DB2 token, you can create reports that show the DB2 activity caused by a CICS transaction.

Monitoring exception and incident data

An *exception* is an event that you should monitor. An exception appears in a report only if it has occurred; reports do not show null counts. A single exception need not be a cause for alarm. An incident is defined as an exception with severity 1, 2, or 3.

The EPDM CICS Performance feature creates exception records for these incidents and exceptions:

- Wait for storage
- Wait for main temporary storage
- Wait for a file string
- Wait for a file buffer
- Wait for an auxiliary temporary storage string
- Wait for an auxiliary temporary storage buffer
- Transaction ABEND
- System ABEND
- Storage violations
- Short-of-storage conditions
- VTAM request rejections
- I/O errors on auxiliary temporary storage
- I/O errors on the intrapartition transient data set

- Autoinstall errors
- MXT reached
- DTB overflow
- Link errors for IRC and ISC
- Journal buffer-full conditions
- CREAD and CWRITE fails (data space problems)
- Local shared pool (LSR) string waits (from A08BKTSW)
- Waits for a buffer in the LSR pool (from A09TBW)
- Errors writing to SMF
- No space on transient-data data set (from A11ANOSP)
- Waits for a transient-data string (from A11STNWT)
- Waits for a transient-data buffer (from A11ATNWT)
- Transaction restarts (from A02ATRCT)
- Maximum number of tasks in a class reached (CMXT) (from A15MXTM)
- Transmission errors (from A06TETE or AUSTETE)

Figure 19 shows an incident report.

```
                                  CICS Incidents
                             DATE: '1993-02-01' to '1993-02-02'

                           Terminal
                           operator  User    Exception           Exception
Sev  Date        Time         ID      ID        ID              description
---  ----------  --------  -------- -------- ----------------- --------------------------
03   1993-02-01  15.42.03  SYSTEM            TRANSACTION_ABEND  CICS TRANSACTION ABEND AZTS
03   1993-02-02  00.00.00  SYSTEM            TRANSACTION_ABEND  CICS TRANSACTION ABEND APCT
03   1993-02-02  17.37.28  SYSTEM            SHORT_OF_STORAGE   CICS SOS IN PAGEPOOL
03   1993-02-02  17.12.03  SYSTEM            SHORT_OF_STORAGE   CICS SOS IN PAGEPOOL

                                       EPDM Report: CICS002
```

Figure 19. Example of an EPDM CICS incidents report

EPDM can pass the exceptions to an Information/Management system.

Unit-of-work reporting

In a CICS multiple region operation (MRO) or intersystem communication (ISC) environment, you can trace a transaction as it migrates from one region (or processor complex) to another and back. The data lets you determine the total resource requirements of the combined transaction as a unit of work, without having to separately analyze the component transactions in each region. The ability to combine the component transactions of an MRO or ISC series makes possible precise resource accounting and chargeback, and capacity and performance analysis.

The CICS UOW Response Times report in Figure 20 on page 107 shows an example of how EPDM presents CICS unit-of-work response times.

```
                        CICS UOW Response Times
                    Time: '09.59.00' to '10.00.00'
                           Date: 1993-03-10


Adjusted
  UOW                                          UOW   Response
 start    Tran  CICS      Program   tran      time
  time     ID    ID        name     count     (sec)
--------  ----  --------  --------  -----  --------
09.59.25  OP22  CICSPROD  DFHAPRT       2     0.436
          OP22  CICSPRDC  OEPCPI22

09.59.26  AP63  CICSPRDE  APPM00        2     0.045
          AP63  CICSPROD  DFHAPRT

09.59.26  ARUS  CICSPROD  DFHAPRT       3     0.158
          CSM5  CICSPRDB  DFHMIR
          ARUS  CICSPRDC  AR49000

09.59.27  CSM5  CICSPRDB  DFHMIR        4     0.639
          CSM5  CICSPRDB  DFHMIR
          MQ01  CICSPROD  DFHAPRT
          MQ01  CICSPRDD  CMQ001

 ...

                      EPDM Report: CICS902
```

Figure 20. EPDM CICS UOW response times report

Monitoring availability

Users of CICS applications depend on the availability of several types of resources:

- Central site hardware and the operating system environment in which the CICS region runs
- Network hardware, such as communication controllers, teleprocessing lines, and terminals through which users access the CICS region
- CICS region
- Application programs and data. Application programs can be distributed among several CICS regions.

In some cases, an application depends on the availability of many resources of the same and of different types, so reporting on availability requires a complex analysis of data from different sources. EPDM can help you, because all the data is in one database.

Monitoring SYSEVENT data

If the SYSEVENT option is used, CICS records at the end of each transaction:

- Transaction ID
- Associated terminal ID
- Elapsed time

This is useful when you require only transaction statistics, rather than the detailed information that CMF produces. In many cases, it may be sufficient to process only this data, since RMF records it as part of its SMF type-72 record. Analysis (and even recording) of SMF records from CMF can then be reserved for those

circumstances when the detailed data is needed. Use the EPDM System Performance feature (MVS Performance component) to report on this data.

When running under goal mode in MVS 5.1.0 and later, CICS performance can be reported in workload groups, service classes, and periods. These are a few examples of EPDM reports for CICS in this environment.

```
                        MVSPM Served Service Classes, Overview
                        Sysplex: 'SYSPLEX1' System: MVS_SYSTEM_ID
                        Date: '1994-02-01' Period: 'PRIME'


      Workload Service  Served    No of times         No of    No of times
       group    class   class       served            tx's    served per tx
      -------- -------- -------- ------------  ------------  --------------
      CICS     CICSREGS CICS-1          15227           664            22.9
                        CICS-2           6405           215            29.8
                        CICS-3          24992          1251            20.0
                        CICS-4          87155          1501            58.1
                        CICSTRX         67769          9314             7.3

                                  EPDM Report: MVSPM79
```

Figure 21. Example of an MVSPM served service classes overview report

Figure 21 shows how service classes were served by other service classes. This report is available only when the MVS system is running in goal mode.

```
                                MVSPM Response Time Breakdown, Overview
                                Sysplex: 'SYSPLEX1' Subsystem: SUBSYSTEM
                                Date: '1994-02-01' Period: 'PRIME'


          Service        MVS  Total Activ Ready Idle   Lock  I/O   Conv Distr Local  Netw Syspl Timer Other  Misc
Workload  class          sys  state state state state  wait  wait  wait wait  wait   wait wait  wait  wait   wait
  group   /Period    Ph  ID    (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)
-------- ---------- --- ---- ----- ----- ----- ----- ----- ----- ----- --- ----- ----- ----- ----- ----- ----- -----
CICS     CICS-1  /1 BTE CA0    6.6   0.0   0.0   0.0   0.0   0.0   6.5   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                        C80   29.4   0.0   0.0   0.0   0.0   0.0  14.7   0.0   0.0   0.0   0.0   0.0  14.6   0.0
                        C90    3.8   0.4   1.3   1.5   0.0   0.2   0.5   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                           ----- ----- ----- ----- ----- ----- -----  --- ----- ----- ----- ----- ----- ----- -----
                           *  13.3   0.1   0.5   0.5   0.0   0.1   7.2   0.0   0.0   0.0   0.0   0.0   4.9   0.0


                    /1 EXE CA0   16.0   0.1   0.2   0.1   0.0  15.5   0.0   0.0   0.0   0.0   0.0   0.0   0.1   0.0
                        C80   14.9   0.1   0.1   0.1   0.0   3.7   0.0   0.0   0.0   0.0   0.0   0.0  11.0   0.0
                        C90   14.0   1.6   4.5   4.8   0.0   3.2   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                           ----- ----- ----- ----- ----- ----- -----  --- ----- ----- ----- ----- ----- ----- -----
                           *  14.9   0.6   1.6   1.7   0.0   7.4   0.0   0.0   0.0   0.0   0.0   0.0   3.7   0.0


IMS      IMS-1   /1 EXE CA0   20.7   0.4   0.7   0.0   0.0   0.0  19.6   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                        C80    1.1   0.2   0.1   0.7   0.0   0.1   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                        C90   22.2   5.3  11.9   1.2   0.0   0.2   3.6   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                           ----- ----- ----- ----- ----- ----- -----  ---- ----- ----- ----- ----- ----- ----- -----
                           *  14.7   2.0   4.2   0.6   0.0   0.1   7.8   0.0   0.0   0.0   0.0   0.0   0.0   0.0

                                        EPDM Report: MVSPM73
```

Figure 22. Example of an MVSPM response time breakdown overview report

Figure 22 shows how much the various transaction states contribute to the average response time. This report is available when the MVS system is running in goal mode and when the subsystem is CICS or IMS.

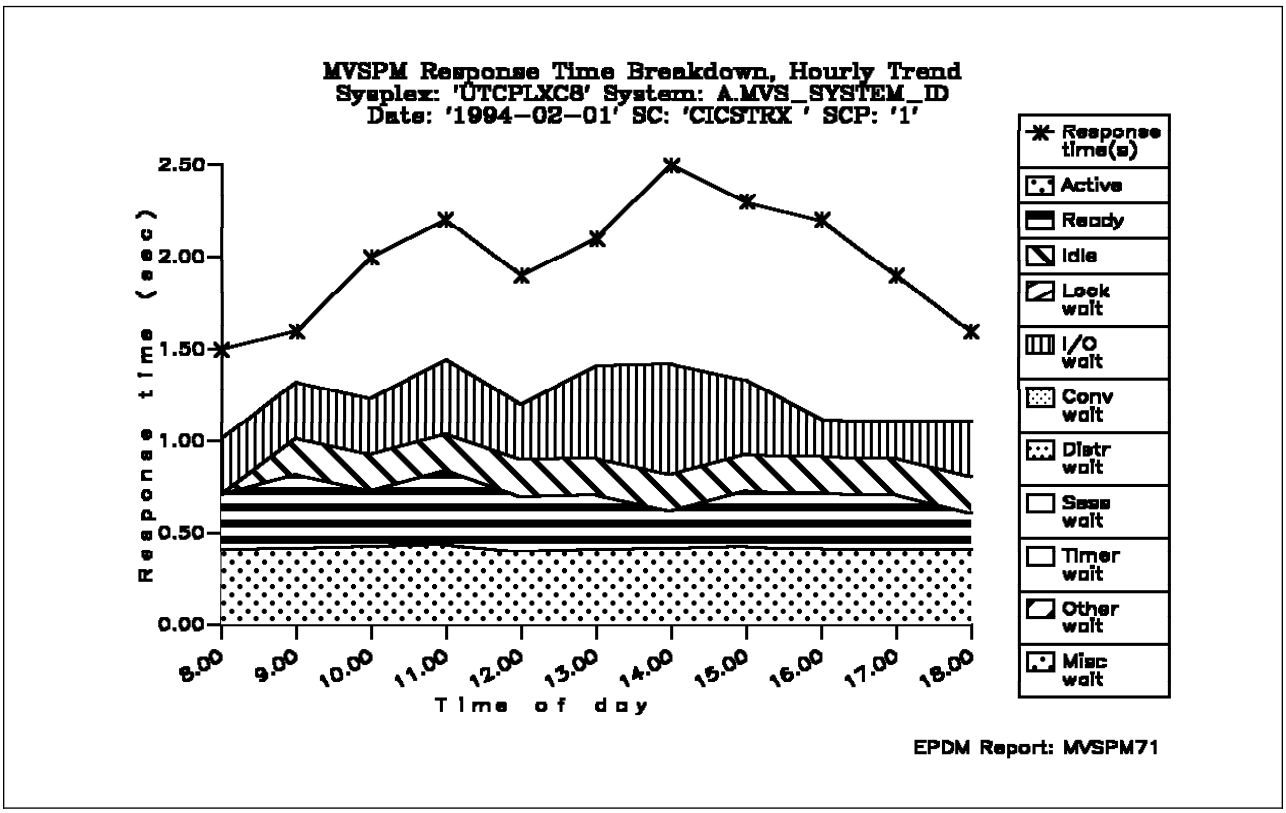


Figure 23. Example of an MVSPM response time breakdown, hourly trend report

Figure 23 shows the average transaction response time trend and how the various transaction states contribute to it. The sum of the different states adds up to the average execution time. The difference between the response time and the execution time is mainly made up of switch time, for example, the time the transactions spend being routed to another region for processing. This report is available when the MVS system is running in goal mode and when the subsystem is CICS or IMS.

Chapter 9. MVS Workload Manager

MVS/ESA 5.1 helps you manage your resources across MVS subsystems. This facility, the MVS workload manager, provides automatic, dynamic, balancing of system resources (central processors and storage) across a sysplex by:

- Adopting a goal-oriented approach
- Gathering real-time data from the subsystems that reflect performance at an individual task level
- Monitoring MVS- and subsystem-level delays and waits that are contributing to overall task execution times
- Dynamically managing the sysplex's resources, using the performance goals, and the real-time performance and delay data, as inputs to system resource management algorithms.

This is particularly significant in a sysplex environment, but is also of value to subsystems running in a single MVS image.

To help you migrate to goal-oriented workload management, you can run any MVS image in a sysplex in **compatibility mode**, using the performance management tuning methods of releases of MVS before MVS/ESA 5.1.

Note: If you use CICSPlex SM to control dynamic transaction routing in a sysplex, you can base its actions on the CICS response time goals of the CICS transactions as defined to the MVS workload manager. For more information, see the *IBM CICSPlex System Manager for MVS/ESA Setup and Administration* manual.

The benefits of using MVS workload manager are:

- Improved performance through MVS resource management

  The improvement is likely to depend on many factors, for example:

  – System hardware configuration
  – The way the system is partitioned
  – Whether CICS subsystems are single or multi-region
  – The spread of types of applications or tasks performed, and the diversity of their profile of operation
  – The extent to which the sysplex workload changes dynamically.

- Improved efficiency of typical MVS sysplexes
  – improved overall capacity
  – Increased work throughput.

- Simplified MVS tuning

  Generally, systems whose operating signature makes attaining or maintaining optimal tuning difficult or time consuming to achieve by current means will tend to obtain the greater benefit.

The main benefit is that you no longer have to continually monitor and tune CICS to achieve optimum performance.

The MVS workload manager produces performance reports that you can use to establish reasonable performance goals and for capacity planning.

MVS workload management terms

The following terms are used in the description of MVS workload management:

classification rules
The rules workload management and subsystems use to assign a service class and, optionally, a reporting class to a work request (transaction). A classification rule consists of one or more of the following work qualifiers:

- subsystem type
- subsystem instance
- userid
- accounting information
- transaction name
- transaction class
- source LU, NETID, and LU name.

compatibility mode
A workload management mode for an MVS image in a sysplex using the pre-workload management MVS performance tuning definitions from the IEAICSxx and IEAIPSxx members of the SYS1.PARMLIB library.

goal mode
A workload management mode for an MVS image in a sysplex using an MVS workload management service definition to automatically and dynamically balance its system resources according to the active service policy for the sysplex.

report class
Work for which reporting information is collected separately. For example, you can have a report class for information combining two different service classes, or a report class for information on a single transaction.

service class
A subset of a workload having the same service goals or performance objectives, resource requirements, or availability requirements. For workload management, you assign a service goal to a service class.

service definition
An explicit definition of all the workloads and processing capacity in a sysplex. A service definition includes service policies, workloads, service classes, resource groups, and classification rules.

service policy
A set of performance goals for all MVS images using MVS workload manager in a sysplex. There can be only one active service policy for a sysplex, and all subsystems in goal mode within that sysplex process towards that policy. However, you can create several service policies, and switch between them to cater for the different needs of different processing periods.

workload
Work to be tracked, managed and reported as a unit. Also, a group of service classes.

workload management mode
The mode in which workload management manages system resources in an MVS image within a sysplex. The mode can be either compatibility mode or goal mode.

Span of workload manager operation

MVS workload manager operates across a sysplex. You can run each MVS image in the sysplex in either goal mode or compatibility mode. However, there can be only one active service policy for all MVS images running in goal mode in a sysplex.

All CICS regions (and other MVS subsystems) running on an MVS image with MVS workload manager are subject to the effects of workload management.

If the CICS workload involves non-CICS resource managers, such as DB2 and DBCTL, CICS can pass information through the resource manager interface (RMI[1]) to enable MVS workload manager to relate the part of the workload within the non-CICS resource managers to the part of the workload within CICS.

CICS does not pass information across ISC links to relate the parts of the task execution thread on either side of the ISC link. If you use tasks that communicate across ISC links, you must define separate performance goals, and service classes, for the parts of the task execution thread on each side of the ISC link. These rules apply to ISC links that are:

- Within the same MVS image (so called “intrahost ISC”)
- Between MVS images in the same sysplex (perhaps for compatibility reasons)
- Between MVS images in different sysplexes.

If you use tasks that communicate across ISC links between two sysplexes, the separate performance goals are defined in the active service policy for each sysplex.

Defining performance goals

You can define performance goals, such as internal response times, for CICS (and other MVS subsystems that comprise your workload). As an alternative to defining your own goals, you can use “discretionary goals”—the workload manager decides how best to run work for which this type of goal is specified. You can define goals for:

- Individual CICS regions
- Groups of transactions running under CICS
- Individual transactions running under CICS
- Transactions associated with individual userids
- Transactions associated with individual LU names.

The service level administrator defines your installation’s performance goals based on business needs and current performance. The complete definition of workloads and performance goals is called a **service definition**. You may already have this kind of information in a service level agreement (SLA).

---

[1] The CICS interface modules that handle the communication between a task-related user exit and the resource manager are usually referred to as the resource manager interface (RMI) or the task-related user exit (TRUE) interface.

You should record the details of your planned service definition on worksheets, as described in the *MVS/ESA Planning: Workload Management* manual. MVS/ESA 5.1 provides an ISPF panel-based application for setting up and adjusting the service definition.

Workload management also collects performance and delay data for work defined by the service definition; this data can be used by reporting and monitoring products, such as the Resource Measurement Facility (RMF), the Service Level Reporter (SLR), Enterprise Performance Data Manager/MVS (EPDM), or vendor products.

Before you set goals for CICS work, you can determine CICS current response times by running CICS in compatibility mode with an arbitrary goal. For this purpose, use the SRVCLASS parameter, provided by MVS/ESA 5.1 in the installation control specification (ICS). This parameter lets you associate a service class with a report performance group, to be run in compatibility mode. You would then:

1. Define a service policy, with a default service class, or classes, for your CICS work, and specify an arbitrary response time goal (say 3 seconds)
2. Define classification rules for the service class or classes
3. Install the service definition
4. Activate the service policy in compatibility mode.

The average response time for work within the service classes is reported under the report performance group in the RMF Monitor I workload activity report.

This information helps you to set realistic goals for running your CICS work when you switch to goal mode. The reporting data produced by RMF reports:

- Is organized by service class
- Contains reasons for any delays that affect the response time for the service class (for example, because of the actions of a resource manager or an I/O subsystem).

From the reported information, you may be able to determine configuration changes to improve performance.

Example of using SRVCLASS parameter of IEAICSxx: To obtain 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, comprising the following:

```
Service Policy Name . . . : CICSTEST
Description . . . . . : Migration (compatibility) mode
```

  - A workload definition, in which to define the required service class:

```
Workload Name  . . . . . : CICSALL
Description  . . . . . . . . CICSTEST migration workload
```

– A service class for all CICS transactions:

```
Service Class Name . . . . . : CICSALL
Description  . . . . . . . . . All CICS transactions
Workload Name  . . . . . . . . CICSALL
        ---Period---  ----------------------Goal---------------------
Action  #  Duration   Imp.  Description
  __    1              1    Average response time of 00:00:03.000
```

Note: It does not matter what goal you specify, since it is not used in compatibility mode, but it cannot be discretionary.

– Specify the name of the service class under the classification rules for the CICS subsystem:

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

- In your ICS member in SYS1.PARMLIB (IEAICS*xx*), specify:

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

- Install the workload definition in the coupling facility.
- Activate the test service policy, either by using options provided by the WLM ISPF application, or by issuing the following MVS command:

  ```
  VARY WLM,POLICY=CICSTEST
  ```

You receive response time information about CICS transactions in the RMF Monitor I Workload Activity Report under report performance group 100. For more information about defining performance goals and the use of SRVCLASS, see the *MVS/ESA Planning: Workload Management* manual.

Service definitions

You define one service definition for each sysplex. A service definition consists of:

Service policies
: You can have one or more service policies, which are a named set of performance goals meant to cover a certain operating period.

  If you have varying performance goals, you can define several service policies.

  You can activate only one service policy at a time for the whole sysplex, and, when appropriate, switch to another policy.

Workloads
: A workload comprise units of work that share some common characteristics that makes it meaningful for an installation to manage or monitor as a group. For example, all CICS work, or all CICS order entry work, or all CICS development work.

  A workload is made up of one or more service classes.

Service classes
: These are categories of work, within a workload, to which you can assign performance goals.

You can create service classes for groups of work with similar:

- Performance goals

  You can assign the following performance goals to the service classes:

  Response time
  : You can define an average response time (the amount of time required to complete the work) or a response time with percentile (a percentage of work to be completed in the specified amount of time).

  Discretionary
  : You can specify that the goal is discretionary for any work for which you do not have specific goals.

  Velocity
  : For work not related to transactions, such as batch jobs and started tasks. For CICS regions started as started tasks, a velocity goal applies only during start-up.

  Notes:

  1. For service classes for CICS transactions, you cannot define velocity performance goals, discretionary goals, or multiple performance periods.
  2. For service classes for CICS regions, you cannot define multiple performance periods.

- Business importance to the installation

  You can assign an importance to a service class, so that one service class goal is recognized as more important than other service class goals. There are five levels of importance, numbered, from highest to lowest, 1 to 5.

You can also create service classes for started tasks and JES, and can assign resource groups to those service classes. You can use such service classes to manage the workload associated with CICS as it starts up, but before CICS transaction-related work begins. (Note that when you define CICS in this way, the address space name is specified as TN, for the task or JES “transaction” name.)

There is a default service class, called SYSOTHER. It is used for CICS transactions for which MVS workload management cannot find a matching service class in the classification rules—for example, if the couple data set becomes unavailable.

Classification rules
: These rules determine how to associate incoming work with a service class. Optionally, the classification rules can assign incoming work to a report class, for grouping report data.

  There is one set of classification rules for each service definition. The classification rules apply to every service policy in the service definition; so there is one set of rules for the sysplex.

  You should use classification rules for every service class defined in your service definition.

  Classification rules categorize work into service classes and, optionally, report classes, based on work qualifiers. You set up classification rules for each MVS

subsystem type that uses workload management. The work qualifiers that CICS can use (and which identify CICS work requests to workload manager) are:

LU	LU name
LUG	LU name group
SI	Subsystem instance (VTAM applid)
SIG	Subsystem instance group
TN	Transaction identifier
TNG	Transaction identifier group
UI	Userid
UIG	Userid group.

Notes:

1. You should consider defining workloads for terminal-owning regions only. Work requests do not normally originate in an application-owning region. They (transactions) are normally routed to an application-owning region from a terminal-owning region, and the work request is classified in the terminal-owning region. In this case, the work is not reclassified in the application-owning region.

   If work orginates in the application-owning region it is classified in the application-owning region; normally there would be no terminal.

2. You can use identifier group qualifiers to specify the name of a group of qualifiers; for example, GRPACICS could specify a group of CICS tranids, which you could specify on classification rules by TNG GRPACICS. This is a useful alternative to specifying classification rules for each transaction separately.

You can use classification groups to group disparate work under the same work qualifier—if, for example, you want to assign it to the same service class.

You can set up a hierarchy of classification rules. When CICS receives a transaction, workload manager searches the classification rules for a matching qualifier and its service class or report class. Because a piece of work can have more than one work qualifier associated with it, it may match more than one classification rule. Therefore, the order in which you specify the classification rules determines which service classes are assigned.

Note: You are recommended to keep classification rules simple.

Example of using classification rules: As an example, you might want all CICS work to go into service class CICSB except for the following:

- All work from LU name S218, except the PAYR transaction, is to run in service class CICSA
- Work for the PAYR transaction (payroll application) entered at LU name S218 is to run in service class CICSC.
- All work from terminals other than LU name S218, and whose LU name begins with S2, is to run in service class CICSD.

You could specify this by the following classification rules:

```
Subsystem Type . . . . . CICS

    -------Qualifier-----------             -------Class--------
    Type       Name     Start                Service     Report
                                  DEFAULTS: CICSB       ________
1   LU         S218                         CICSA       ________
2     TN         PAYR                       CICSC       ________
1   LU         S2*                          CICSD       ________
```

Note: In this classification, the PAYR transaction is nested as a sub-rule under the classification rule for LU name S218, indicated by the number 2, and the indentation of the type and name columns.

Consider the effect of these rules on the following work requests:

	Request 1	Request 2	Request 3	Request 4
LU name	S218	A001	S218	S214
Transaction ..	PAYR	PAYR	DEBT	ANOT

- For request 1, the work request for the payroll application runs in service class CICSC. This is because the request is associated with the terminal with LU name S218, and the TN—PAYR classification rule specifying service class CICSC is nested under the LU—S218 classification rule qualifier.
- For request 2, the work request for the payroll application runs in service class CICSB, because it is ***not*** associated with LU name S218, nor S2*, and there are no other classification rules for the PAYR transaction. Likewise, any work requests associated with LU names that do not start with S2 run in service class CICSB, as there are classification rules for LU names S218 and S2* only.
- For request 3, the work request for the DEBT transaction runs in service class CICSA, because it is associated with LU name S218, and there is no DEBT classification rule nested under the LU—S218 classification rule qualifiers.
- For request 4, the work request for the ANOT transaction runs in service class CICSD, because it is associated with an LU name starting S2, but not S218.

However, if the classification rules were specified as:

```
1 TN         PAYR                       CICSA       ________
1 LU         S218                       CICSA       ________
2   TN       PAYR                       CICSC       ________
1 LU         S2*                        CICSD       ________
```

the PAYR transaction would always run in service class CICSA, even if it were associated with LU name S218.

Guidelines for classifying CICS transactions: For RMF to provide meaningful Workload Activity Report data it is suggested that you use the following guidelines when defining the service classes for CICS transactions. In the same service class:

1. Do not mix CICS-supplied transactions with user transactions
2. Do not mix routed with non-routed transactions
3. Do not mix conversational with pseudo-conversational transactions
4. Do not mix long-running and short-running transactions.

Using a service definition base: To minimize the amount of data you need to enter into the ISPF workload application, you use a **service definition base**. When you set up your service definition, you identify the workloads, the service classes, and their goals, based on your performance objectives. Then you define classification rules. This information makes up the service definition base. The base contains workloads, service classes, resource groups, report classes, and classification rules.

All workloads, service classes, and classification rules defined in a service definition base apply to every policy that you define. You should use classification rules for every service class defined in your service definition. If you do not have any other business requirements to modify a service goal or a resource group from the service definition base, you can run an installation with one policy.

MVS performance definitions

Running CICS on MVS/ESA 5.1 or later, enables you to use the MVS workload management facility that MVS/ESA 5.1 provides for managing sysplex resources across MVS subsystems, in parallel with the existing system resource management facilities.

For information about MVS workload management, see the MVS publication *Planning: Workload Management*, GC28-1493.

If you want to use the MVS workload manager facility, you should:

1. Implement workload management on the MVS images that the CICS workload is to run on, as outlined in "Implementing MVS workload management."
2. Ensure that CICS performance parameters correspond to the policies defined for MVS workload management, as outlined in "Matching CICS performance parameters to service policies" on page 121.

If you have not installed MVS/ESA 5.1, or do not want to use the MVS workload management facility, you should review your MVS performance definitions to ensure that they are still appropriate for CICS/ESA 4.1. To do this, review parameters in the IEAICS and IEAIPS members of the MVS PARMLIB library. For more information about these MVS performance definitions, see the *MVS/ESA Initialization and Tuning Guide*.

Implementing MVS workload management

The task of implementing MVS workload management is part of the overall task of planning for, and installing, MVS/ESA 5.1.

Implementing MVS workload management generally involves the following steps:

1. Establish your workloads.
2. Set your business priorities.
3. Understand your performance objectives.

4. Define critical work.
5. Define performance objectives based on current:
   - Business needs
   - Performance:
     - Reporting and monitoring products
     - Capacity planning tools
     - IEAICS and IEAIPS parameters.
6. Get agreement for your workload performance objectives.
7. Specify a service level agreement or performance objectives.
8. Specify an MVS WLM service definition using the information from step 7.

   Note: It is helpful at this stage to record your service definition in a form that will help you to enter it into the MVS workload manager ISPF application. You are recommended to use the worksheets provided in the MVS publication *Planning: Workload Management*.

9. Install MVS/ESA 5.1.
10. Set up a sysplex with a single MVS image, and run in workload manager compatibilty mode.
11. Upgrade your existing XCF couple data set.
12. Start the MVS workload manager ISPF application, and use it in the following steps.
13. Allocate and format a new couple data set for workload management. (You can do this from the ISPF application.)
14. Define your service definition.
15. Install your service definition on the couple data set for workload management.
16. Activate a service policy.
17. Switch the MVS image into goal mode.
18. Start up a new MVS image in the sysplex. (That is, attach the new MVS image to the couple data set for workload management, and link it to the service policy.)
19. Switch the new MVS image into goal mode.
20. Repeat steps 18 and 19 for each new MVS image in the sysplex.

Notes:

1. CICS/ESA 4.1 support for MVS workload manager is initialized automatically during CICS startup.
2. All CICS regions (and other MVS subsystems) running on an MVS image with MVS workload management are subject to the effects of workload manager.

Matching CICS performance parameters to service policies

You must ensure that the CICS performance parameters are compatible with the workload manager service policies used for the CICS workload.

In general, you should define CICS performance objectives to the MVS workload manager first, and observe the effect on CICS performance. Once the MVS

workload manager definitions are working correctly, you can then consider tuning the CICS parameters to further enhance CICS performance. However, you should use CICS performance parameters as little as possible.

Performance attributes that you might consider using are:

- Transaction priority, passed on dynamic transaction routing. (Use prioritization carefully, if at all.) The priority assigned by the CICS dispatcher must be compatible with the task priority defined to MVS workload manager.
- Maximum number of concurrent user tasks for the CICS region.
- Maximum number of concurrent tasks in each transaction class.

Activating CICS support for MVS workload manager

CICS/ESA 4.1 support for MVS workload manager is initialized automatically during CICS startup.

Customer-written resource managers and other non-CICS code which is attached to CICS via the RMI must be modified to provide workload manager support, if workload manager is to work correctly for CICS-based tasks which cross the RMI into such areas.

Dynamic transaction routing enhancements

These enhancements allow a dynamic transaction routing program to keep track of CICS-initiated transactions that abend in an application-owning region (AOR). This function is exploited by the workload balancing routines in CICSPlex System Manager (SM) Release 1.

Normally, a dynamic transaction program would have to be written to take advantage of dynamic transaction routing. However, with a CICSPlex SM product to manage your CICSplex, you need not do so. CICSplex SM provides a dynamic routing program that supports both workload balancing and workload separation. The CICSPlex SM, through its user interface, is told which TORs and AORs in the CICSplex can participate in dynamic transaction routing, and define any affinities that govern the AORs to which particular transactions must be routed. The output from the Transaction Affinities Utility can be used directly by CICSPlex SM. For more information about CICSPlex SM, see the *CICSPlex SM Concepts and Planning* manual.

There are no special requirements to be able to use the dynamic transaction routing mechanism, but it offers the user the following:

- A dynamic transaction routing program to make more intelligent routing decisions; for example, based on workload goals.
- CICS can provide improved support for MVS goal-oriented workload management.
- Make it easier to use a global temporary storage owning region in the MVS sysplex environment, which avoids intertransaction affinity that can occur with the use of local temporary storage queues.
- Can be used by CICSPlex SM to provide intelligent routing in a CICSPlex that has at least one terminal-owning region linked to multiple application-owning regions.

Requirements for MVS workload management

To use MVS workload management you need the following software:

- MVS/ESA System Product (MVS/ESA SP) - JES2 Version 5 Release 1 or a later, upward-compatible, release
- MVS/ESA System Product (MVS/ESA SP) - JES3 Version 5 Release 1 or a later, upward-compatible, release

For MVS workload manager operation across the CICS task-related user exit interface to other subsystems, such as DB2 and DBCTL, you need the appropriate releases of these products.

For more information about requirements for MVS workload management see the following manuals: *MVS/ESA Planning : Workload Management*, and *MVS/ESA Planning : Sysplex Manager*.

Resource usage

The CICS function for MVS workload management incurs negligible impact on CICS storage.

Chapter 10. Understanding RMF workload manager data

This chapter provides an explanation of CICS-related data from an RMF workload activity report. RMF provides data for subsystem **work managers** that support workload management. In MVS/ESA 5.1, these are IMS and CICS.

This chapter includes a discussion of some possible data that may be reported for CICS and IMS, and provides some possible explanations for the data. Based on this discussion and the explanations, you may decide to alter your service class definitions. In some cases, there may be some actions that you can take, in which case you can follow the suggestion. In other cases, the explanations are provided only to help you better understand the data. For more information about using RMF, see the *RMF User's Guide*.

Explanation of terms used in RMF reports

For those readers familiar with CICS and CICS terminology, it might help to relate some of the terms used in an RMF activity report to the more familiar CICS terms. For example, some of terms in the RMF report can be equated with CEMT INQUIRE TASK terms.

These explanations are given for two main sections of the reports:

- The response time breakdown in percentage section
- The state section, covering switched time.

The response time breakdown in percentage section

The "Response time breakdown in percentage" section of the RMF report contains the following headings:

ACTIVE The percentage of response time accounted for by tasks currently executing in the region—tasks shown as **Running** by the CEMT INQUIRE TASK command.

READY The percentage of response time accounted for by tasks that are not currently executing but are ready to be dispatched—tasks shown as **Dispatchable** by the CEMT INQUIRE TASK command.

IDLE The percentage of response time accounted for by a number of instances or types of CICS tasks:

- Tasks waiting on a principal facility (for example, conversational tasks waiting for a response from a terminal user)
- The terminal control (TC) task, CSTP, waiting for work
- The interregion controller task, CSNC, waiting for transaction routing requests
- CICS system tasks, such as CSSY, waiting for work.

A CEMT INQUIRE TASK command would show any of these user tasks as **Suspended**, as are the CICS system tasks.

WAITING FOR
The percentage of response time accounted for by tasks that are not currently executing and are not ready to be dispatched—shown as **Suspended** by the CEMT INQUIRE TASK command.

The WAITING FOR main heading is further broken down into a number of subsidiary headings. Where applicable, for waits other than those described for the IDLE condition described above, CICS interprets the cause of the wait, and records the 'waiting for' reason in the WLM performance block.

The waiting-for terms used in the RMF report equate to the WLM_WAIT_TYPE parameter on the SUSPEND, WAIT_OLDC, WAIT_OLDW, and WAIT_MVS calls used by the dispatcher, and the SUSPEND and WAIT_MVS calls used in the CICS XPI. These are shown as follows (with the CICS WLM_WAIT_TYPE term, where different from RMF, in parenthesis):

Term	Description
LOCK	Waiting on a lock. For example, waiting for: • A lock on CICS resource • A record lock on a recoverable VSAM file • Exclusive control of a record in a BDAM file • An application resource that has been locked by an EXEC CICS ENQ command.
I/O (IO)	Waiting for an I/O request or I/O related request to complete. For example: • File control, transient data, temproary storage, or journal I/O. • Waiting on I/O buffers or VSAM strings.
CONV	Waiting on a conversation between work manager subsystems. This information is further analyzed under the SWITCHED TIME heading.
DIST	Not used by CICS.
LOCAL (SESS_LOCALMVS)	Waiting on the establishment of a session with another CICS region in the same MVS image in the sysplex.
SYSPL (SESS_SYSPLEX)	Waiting on establishment of a session with another CICS region in a different MVS image in the sysplex.
REMOT (SESS_NETWORK)	Waiting on the establishment of an ISC session with another CICS region (which may, or may not, be in the same MVS image).
TIMER	Waiting for a timer event or an interval control event to complete. For example, an application has issued an EXEC CICS DELAY or EXEC CICS WAIT EVENT command which has yet to complete.
PROD (OTHER_PRODUCT)	Waiting on another product to complete its function; for example, when the work request has been passed to a DB2 or DBCTL subsystem.
MISC	Waiting on a resource that does not fall into any of the other categories.

The state section

The state section covers the time that transactions are “switched” to another CICS region:

SWITCHED TIME
The percentage of response time accounted for by tasks in a TOR that are waiting on a conversation across an intersystem communication link (MRO or ISC). This information provides a further breakdown of the response time shown under the CONV heading.

The SWITCHED TIME heading is further broken down into a number of subsidiary headings, and covers those transactions that are waiting on a conversation. These are explained as follows:

LOCAL The work request has been switched, across an MRO link, to another CICS region in same MVS image.

SYSPL The work request has been switched, across an XCF/MRO link, to another CICS region in another MVS image in the sysplex.

REMOT The work request has been switched, across an ISC link, to another CICS region (which may, or may not, be in the same MVS image).

Interpreting the RMF workload activity data

Figure 25 on page 128 shows an example of the CICS state section of an RMF Monitor I workload activity report. It is based on an example hotel reservations service class.

The text following the figure explain how to interpret the fields.

RMF reporting intervals

An RMF workload activity report contains “snapshot data,” which is data collected over a relatively short interval. The data for a given work request (CICS transaction) in an MRO environment is generally collected for more than one CICS region, which means there can be some apparent inconsistencies between the execution (EXE) phase and the begin to end (BTE) data in the RMF reports. This is caused by the end of a reporting interval occurring at a point when work has completed in one region but not yet completed in an associated region.

For example, an AOR can finish processing transactions, the completion of which are included in the current reporting interval, whilst the TOR may not complete its processing of the same transactions during the same interval.

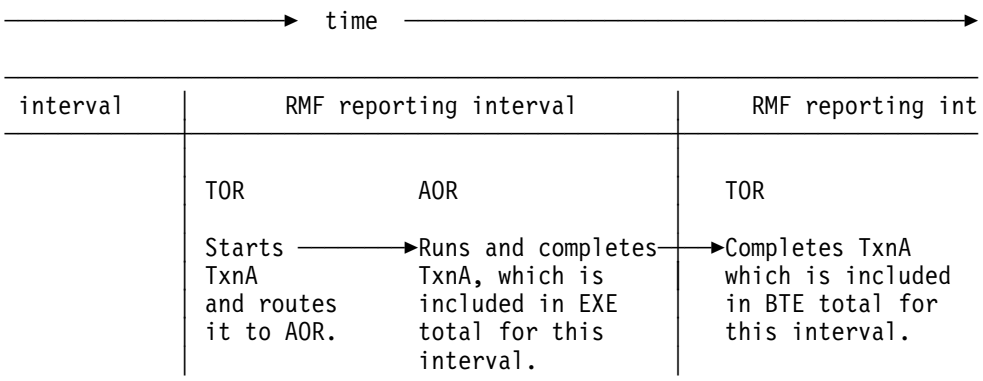


Figure 24. Illustration of snapshot principle for RMF reporting intervals

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSHR RESOURCE GROUP=*NONE PERIOD=1 IMPORTANCE=HIGH

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG        0.00  ACTUAL             000.00.00.114
MPL        0.00  QUEUED             000.00.00.036
ENDED       216  EXECUTION          000.00.00.078
END/SEC    0.24  STANDARD DEVIATION 000.00.00.270
#SWAPS        0
EXECUTD     216

           -------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE------------------  ----STATE------
SUB   P  TOTAL  ACTIVE  READY IDLE  ------------------------WAITING FOR--------------------  SWITCHED TIME (%)
TYPE                                LOCK I/O  CONV  DIST LOCAL SYSPL  REMOT TIMER  PROD  MISC  LOCAL SYSPL REMOT
CICS BTE  93.4   10.2   0.0   0.0   0.0  0.0  83.3   0.0  0.0   0.0    0.0   0.0    0.0   0.0   83.3  0.0   0.0
CICS EXE  67.0   13.2   7.1   0.0   0.0  0.0   0.0   0.0  0.0   0.0    0.0   0.0   46.7   0.0    0.0  0.0   0.0
```

Figure 25. Hotel Reservations service class

The fields in this RMF report describe an example CICS hotel reservations service class (CICSHR), explained as follows:

ENDED This field shows that 216 hotel reservation transactions completed.

Note: In our example the two phases show the same number of transactions completed, indicating that during the reporting interval all the transactions routed by the TORs (ENDED) were completed by the AORs (EXECUTD) and also completed by the TORs. This will not normally be the case because of the way data is captured in RMF reporting intervals. See “RMF reporting intervals” on page 127.

ACTUAL Shown under TRANSACTION TIME, this field shows the average response time as 0.114 seconds, for the 216 transactions completed in the BTE phase.

CICS This field indicates that the subsystem work manager is CICS.

BTE This field indicates that the data in the row relates to the **begin-to-end** work phase.

CICS transactions are analyzed over two phases: a begin-to-end (BTE) phase, and an execution (EXE) phase.

The begin-to-end phase usually takes place in the terminal owning region (TOR), which is responsible for starting and ending the transaction.

EXE This field indicates that the data in the row relates to the **execution** work phase. The execution phase can take place in an application owning region (AOR) and a resource-owning region such as an FOR. In our example, the 216 transactions were routed by a TOR to another region for execution, such as an AOR (and possibly an FOR).

EXECUTD
This field shows that the AORs completed 216 transactions in the reporting interval.

EXECUTION
Shown under TRANSACTION TIME, this field shows that on average it took 0.078 seconds for the AORs to execute the transactions.

While executing these transactions, CICS records the states the transactions are experiencing. RMF reports the states in the RESPONSE TIME BREAKDOWN IN PERCENTAGE section of the report, with one line for the begin-to-end phase, and another for the execution phase.

The response time analysis for the BTE phase is described as follows:

For BTE Explanation

TOTAL The CICS BTE total field shows that the TORs have information covering 93.4% of the ACTUAL response time, the analysis of which is shown in the remainder of the row.

ACTIVE On average, the work (transactions) was active in the TORs for only about 10.2% of the ACTUAL response time

READY In this phase, the TORs did not detect that any part of the average response time was accounted for by work that was dispatchable but waiting behind other transactions.

IDLE In this phase, the TORs did not detect that any part of the average response time was accounted for by transactions that were waiting for work.

WAITING FOR
Only one field shows a value in the WAITING FOR section—the CONV value (this is typical for a TOR). It indicates that for about 83.3% of the time, the transactions were waiting on a conversation. This is further explained by the SWITCHED TIME data.

SWITCHED TIME
From the SWITCHED TIME % data you can see the reason for the ‘waiting-on-a-conversation’. This is 83.3 % LOCAL, which indicates that the transactions were routed locally to an AOR on the same MVS image.

Note: In the analysis of the BTE phase, the values do not exactly add up to the TOTAL value because of rounding—in our example, 10.2 + 83.3 = 93.5, against a total shown as 93.4.

The response time analysis for the EXE phase is described as follows:

For EXE Explanation

TOTAL The CICS EXE total field shows that the AORs have information covering 67% of the ACTUAL response time.

ACTIVE On average, the work is active in the AOR for only about 13.2% of the average response time.

READY On average the work is ready, but waiting behind other tasks in the region, for about 7.1% of the average response time.

PROD On average, 46.7% of the average response time is spent outside the CICS subsystem, waiting for another product to provide some service to these transactions.

You can't tell from this RMF report what the other product is, but the probability is that the transactions are accessing data through a database manager such as Database Control (DBCTL) or DB2.

Example: very large percentages in the response time breakdown

Figure 26 on page 130 shows an example of a work manager state section for the CICSPROD service class. In the RESPONSE TIME BREAKDOWN IN PERCENTAGE section of the report, both the CICS EXE and the CICS BTE rows show excessively inflated percentages: 78.8K, 183, 1946 and so on.

```
REPORT BY: POLICY=HPTSPOL1 WORKLOAD=PRODWKLD SERVICE CLASS=CICSPROD  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=HIGH

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG        0.00  ACTUAL             000.00.00.111
MPL        0.00  QUEUED             000.00.00.000
ENDED      1648  EXECUTION          000.00.00.123
END/SEC    1.83  STANDARD DEVIATION 000.00.00.351
#SWAPS        0
EXECUTD    1009

           -------------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE----------------------   ---STATE--------
SUB    P  TOTAL  ACTIVE  READY  IDLE    ------------------------WAITING FOR--------------------------   SWITCHED TIME (%)
TYPE                                    LOCK   I/O  CONV  DIST LOCAL SYSPL REMOT  TIMER   PROD  MISC   LOCAL SYSPL REMOT
CICS  BTE  78.8K   183    265   1946    0.0    0.0   235   0.0   0.0   0.0   0.0   0.0    0.0  76.2K    229   0.0  17.9
CICS  EXE   140    91.8   3.1    0.0    0.0    0.1   0.0   0.0   0.0   0.0   0.0   0.0   45.4   0.0    19.6K  0.0   0.0
```

Figure 26. Response Time percentages greater than 100

Possible explanations

There several possible explanations for the unusual values shown in this sample report:

- Long-running transactions
- Never-ending transactions
- Conversational transactions
- Dissimilar work in service class

Long-running transactions

Suppose that, of the total of 1648 transactions, 1647 of them have ended within 0.1 seconds, and one transaction has been running for 5 minutes and is still executing when the RMF interval expires. RMF will show an average response time of 0.111 seconds, and breakdown that response time into the states.

The subsystem, however, recorded a total of 183 seconds (0.111 seconds per transaction times 1647 transactions equals 182.8) plus 300 seconds (5 times 60 seconds for the one transaction running for 5 minutes.) This is 483 seconds-worth of data describing the CICSPROD transactions. When this is divided by the total of 1648 transactions during the interval it gives approximately 0.3 seconds-worth of

data for each completed transaction. This is 3 times the reported average response time, hence RMF reports states that total 300% of the response time.

When such a long transaction completes, it can easily distort the average response time during that interval. RMF reports the standard deviation and distribution of response times around the goal emphasizing when this occurs.

The long running transactions could be either routed or non-routed transactions. Routed transactions are transactions that are routed from a TOR to one or more AORs. Long-running routed transactions could result in many samples of waiting for a conversation (CONV) in the CICS begin-to-end phase, with the AOR's state shown in the execution phase.

Non-routed transactions execute completely in a TOR, and have no execution (CICS EXE) phase data. Non-routed CICS transactions could inflate the ACTIVE or READY data for the CICS BTE phase.

Never-ending transactions

Never-ending transactions differ from long-running transactions in that they persist for the life of a region. For CICS, these could include the IBM reserved transactions such as CSNC and CSSY, or customer defined transactions. Never-ending transactions are reported in a similar way to long-running transactions, as explained above. However, for never-ending CICS transactions, RMF might report large percentages in IDLE, or under TIMER or MISC in the WAITING FOR section.

Conversational transactions

Conversational transactions are considered long-running transactions. CICS marks the state of a conversational transaction as IDLE when the transaction is waiting for terminal input. Terminal input often includes long end-user think time, so you might see very large values in the IDLE state as a percent of response time for completed transactions.

Dissimilar work in the service class

A service class that mixes:

- Customer and IBM transactions,
- Long-running and short-running transactions
- Routed and non-routed transactions
- Conversational and non-conversational transactions.

can expect to have RMF reports showing that the total states sampled account for more than the average response time. This can be expected if the service class is the subsystem default service class. The default is defined in the classification rules as the service class to be assigned to all work in a subsystem not otherwise assigned a service class.

Possible actions

The following are some actions you could take for reports of this type:

Group similar work into the same service classes: Make sure your service classes represent groups of similar work. This could require creating additional service classes. For the sake of simplicity, you may have only a small number of service classes for CICS work. If there are transactions for which you want the

RMF response time breakdown data, consider including them in their own service class.

Do nothing: For service classes representing dissimilar work such as the subsystem default service class, recognize that the response time breakdown could include long-running or never-ending transactions. Accept that RMF data for such service classes does not make much sense.

Example: response time breakdown data is all zero

Figure 27 on page 132 shows an example of a work manager state section for the CICSLONG service class. All data shows a 0.0 value.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSLONG  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=HIGH
                                                    CICS Long Running Internal Trxs

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG        0.00  ACTUAL             000.00.00.000
MPL        0.00  QUEUED             000.00.00.000
ENDED         0  EXECUTION          000.00.00.000
END/SEC    0.00  STANDARD DEVIATION 000.00.00.000
#SWAPS        0
EXECUTD       0

           -------------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE---------------   ---------STATE---
SUB    P   TOTAL  ACTIVE  READY  IDLE  ----------------------------WAITING FOR----------------   SWITCHED TIME (%)
TYPE                                   LOCK  I/O  CONV  DIST LOCAL SYSPL REMOT TIMER PROD MISC   LOCAL SYSPL REMOT
CICS  BTE   0.0     0.0    0.0    0.0  0.0   0.0  0.0   0.0   0.0   0.0   0.0   0.0   0.0  0.0    0.0   0.0   0.0
```

Figure 27. Response time breakdown percentages all 0.0

Possible explanations

There are two possible explanations:

1. No transactions completed in the interval
2. RMF did not receive data from all systems in the sysplex.

No transactions completed in the interval

While a long-running or never-ending transaction is being processed, RMF saves the service class state samples to SMF Type 72 records, (subtype 3). But when no transactions have completed, (and average response time is 0), the calculations to apportion these state samples over the response time result in 0%.

RMF did not receive data from all systems in the sysplex.

The RMF post processor may have been given SMF records from only a subset of the systems running in the sysplex. For example, the report may represent only a single MVS image. If that MVS image has no TOR, its AORs receive CICS transactions routed from another MVS image or from outside the sysplex. Since the response time for the transactions is reported by the TOR, there is no transaction response time for the work, nor are there any ended transactions.

Possible actions

The following are some actions you could take for reports of this type:

Do nothing: You may have created this service class especially to prevent the state samples of long running transactions from distorting data for your production work. In this case there is no action to take.

Combine all SMF records for the sysplex: The state data is contained in the SMF records. If you combine the data from an MVS image that doesn't have a TOR with another MVS image that does, the state data from the two MVS images is analyzed together by RMF. This ensures that the response time distribution data is no longer reported as zeros.

Example: execution time greater than response time

Figure 28 on page 133 shows an example of a work manager state section for the CICSPROD service class. In the example, there are 1731 ENDED transactions yet the EXECUTD field shows that only 1086 have been executed. The response time (ACTUAL field) shows 0.091 seconds as the average of all 1731 transactions, while the AORs can only describe the execution of the 1086 they participated in, giving an execution time of 0.113.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD  SERVICE CLASS=CICSPROD RESOURCE GROUP=*NONE   PERIOD=1 IMPORTANCE=HIGH
                                                                    CICS Trans not classified singly

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG        0.00  ACTUAL             000.00.00.091
MPL        0.00  QUEUED             000.00.00.020
ENDED      1731  EXECUTION          000.00.00.113
END/SEC    1.92  STANDARD DEVIATION 000.00.00.092
#SWAPS        0
EXECUTD    1086
```

Figure 28. Execution time greater than response time

Possible explanation

The situation illustrated by this example could be explained by the service class containing a mixture of routed and non-routed transactions. In this case, the AORs have recorded states which account for more time than the average response time of all the transactions. The response time breakdown shown by RMF for the execution phase of processing can again show percentages exceeding 100% of the response time.

Possible actions

Define routed and non-routed transactions in different service classes.

Example: large SWITCH LOCAL Time in CICS execution phase

Figure 29 on page 134 shows a work manager state data section for a CICSPROD service class. The SWITCH LOCAL time in the response time breakdown section shows a value of 6645.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSPROD  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=HIGH

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG        0.00  ACTUAL             000.00.00.150
MPL        0.00  QUEUED             000.00.00.039
ENDED      3599  EXECUTION          000.00.00.134
END/SEC    4.00  STANDARD DEVIATION 000.00.00.446
#SWAPS        0
EXECUTD    2961

           -------------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE-----------------   ------STATE------
SUB    P   TOTAL ACTIVE READY  IDLE ----------------------------WAITING FOR---------------------   SWITCHED TIME (%)
TYPE                                LOCK  I/O  CONV  DIST  LOCAL SYSPL REMOT TIMER  PROD  MISC   LOCAL SYSPL REMOT
CICS  BTE  26.8K  75.1  98.4    659   0.0   0.3   154  0.0    0.0   0.0   0.0   0.0   0.0   25.8K  149   0.0   7.8
CICS  EXE   93.7  38.6   5.6    0.0   0.0   0.1   0.0  0.0    0.0   0.0   0.0   0.0   49.4   0.0  6645   0.0   0.0
```

Figure 29. High SWITCH time in a CICS execution environment

Possible explanations

This situation can be explained by instances of distributed transaction processing

If, while executing a transaction, an AOR needs to function ship a request to another region (for example, to a file-owning or queue-owning region), the execution time reported in the RMF report for the AOR (the CICS EXE field) includes the time spent in that other region.

However, if a program initiates distributed transaction processing to multiple back-end regions, there can be many AORs associated with the original transaction. Each of the multiple back-end regions can indicate they are switching control back to the front-end region (SWITCH LOCAL). Thus, with a 1-many mapping like this, there are many samples of the execution phase indicating switched requests—long enough to exceed 100% of the response time of other work completing in the service class.

Possible actions

None.

Example: fewer ended transactions with increased response times

The RMF workload activity report shows increased response times, and a decrease in the number of ended transactions.

Possible explanation

This situation could be caused by converting from ISC to MRO between the TOR and the AOR.

When two CICS regions are connected via a VTAM intersystem communication (ISC) links, the perspective from a WLM viewpoint is that they behave differently from when they are connected via multiregion (MRO) option. One key difference is

that, with ISC, both the TOR and the AOR are receiving a request from VTAM, so each believes it is starting and ending a given transaction. So for a given user request routed from the TOR via ISC to an AOR, there would be 2 completed transactions.

Let's assume they have response times of 1 second and .75 seconds respectively, giving for an average of .875 seconds. When the TOR routes via MRO, the TOR will describe a single completed transaction taking 1 second (in a begin-to-end phase), and the AOR will report it's .75 seconds as execution time. Therefore, converting from an ISC link to an MRO connection, for the same workload, could result in 1/2 the number of ended transactions and a corresponding increase in the response time reported by RMF.

Possible action

Increase CICS transaction goals prior to your conversion to an MRO connection.

Part 3. Analyzing the performance of a CICS system

This part gives an overview of performance analysis, identifies performance constraints, and describes various techniques for performance analysis.

Chapter 11. Overview of performance analysis

There are four main uses for performance analysis:

1. You currently have no performance problems, but you simply want to adjust the system to give better performance, and you are not sure where to start.
2. You want to characterize and calibrate individual stand-alone transactions as part of the documentation of those transactions, and for comparison with some future time when, perhaps, they start behaving differently.
3. A system is departing from previously identified objectives, and you want to find out precisely where and why this is so. Although an online system may be operating efficiently when it is installed, the characteristics of the system usage may change and the system may not run so efficiently. This inefficiency can usually be corrected by adjusting various controls. At least some small adjustments usually have to be made to any new system as it goes live.
4. A system may or may not have performance objectives, but it appears to be suffering severe performance problems.

If you are in one of the first two categories, you can skip this chapter and the next and go straight to Chapter 13, " CICS performance analysis" on page 159.

If the current performance does ***not*** meet your needs, you should consider tuning the system. The basic rules of tuning are:

1. Identify the major constraints in the system.
2. Understand what changes could reduce the constraints, possibly at the expense of other resources. (Tuning is usually a trade-off of one resource for another.)
3. Decide which resources could be used more heavily.
4. Adjust the parameters to relieve the constrained resources.
5. Review the performance of the resulting system in the light of:
   - Your existing performance objectives
   - Progress so far
   - Tuning effort so far.
6. Stop if performance is acceptable; otherwise do one of the following:
   - Continue tuning
   - Add suitable hardware capacity
   - Lower your system performance objectives.

The tuning rules can be expressed in flowchart form as follows:

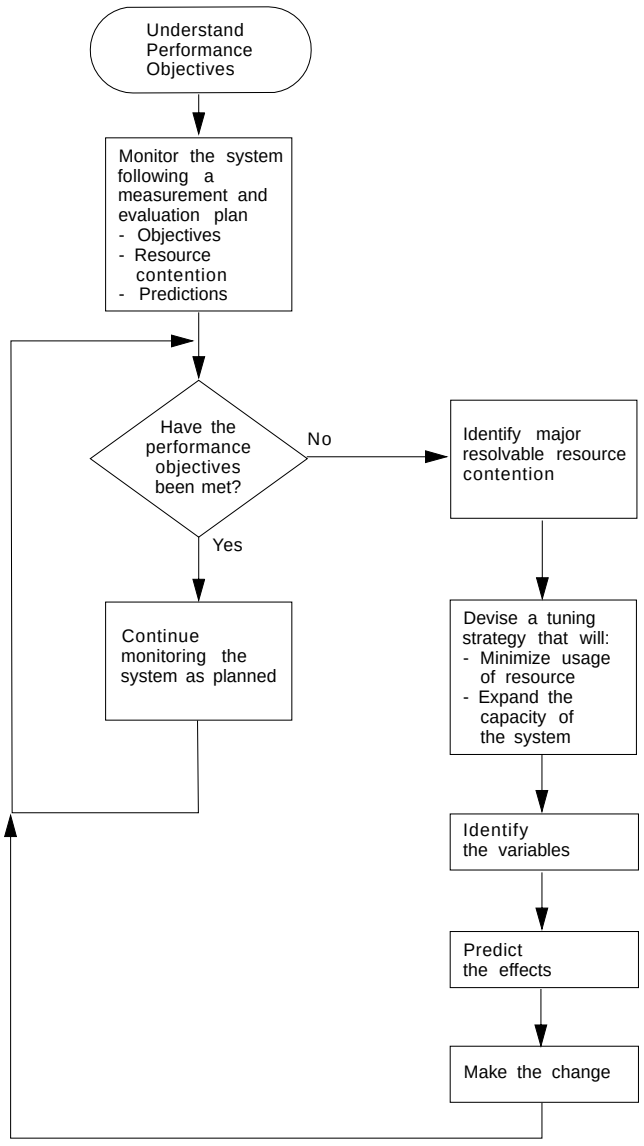


Figure 30. Flowchart to show rules for tuning performance

Establishing a measurement and evaluation plan

For some installations, a measurement and evaluation plan might be suitable. A measurement and evaluation plan is a structured way to measure, evaluate, and monitor the system's performance. By taking part in setting up this plan, the users, user management, and your own management will know how the system's performance is to be measured. In addition, you will be able to incorporate some of their ideas and tools, and they will be able to understand and concur with the plan, support you and feel part of the process, and provide you with feedback.

The implementation steps for this plan are:

1. Devise the plan
2. Review the plan
3. Implement the plan
4. Revise and upgrade the plan as necessary.

Major activities in using the plan are:

- Collect information periodically to determine:
  - Whether objectives have been met
  - Transaction activity
  - Resource utilization.
- Summarize and analyze the information. For this activity:
  - Plot volumes and averages on a chart at a specified frequency
  - Plot resource utilization on a chart at a specified frequency
  - Log unusual conditions on a daily log
  - Review the logs and charts weekly.
- Make or recommend changes if objectives have not been met.
- Relate past, current, and projected:
  - Transaction activity
  - Resource utilization

  to determine:
  - If objectives continue to be met
  - When resources are being used beyond an efficient capacity.
- Keep interested parties informed by means of informal reports, written reports, and monthly meetings.

A typical measurement and evaluation plan might include the following items as objectives, with statements of recording frequency and the measurement tool to be used:

- Volume and response time for each department
- Network activity:
  - Total transactions
  - Tasks per second
  - Total by transaction type
  - Hourly transaction volume (total, and by transaction).

- Resource utilization examples:
  - DSA utilization
  - Processor utilization with CICS
  - Paging rate for CICS and for the system
  - Channel utilization
  - Device utilization
  - Data set utilization
  - Line utilization.
- Unusual conditions:
  - Network problems
  - Application problems
  - Operator problems
  - Transaction count for entry to transaction classes
  - SOS occurrences
  - Storage violations
  - Device problems (not associated with the communications network)
  - System outage
  - CICS outage time.

Investigating the overall system

Always start by looking at the overall system before you decide that you have a specific CICS problem. The behavior of the system as a whole is usually just as important. You should check such things as total processor usage, DASD activity, and paging.

Performance degradation is often due to application growth that has not been matched by corresponding increases in hardware resources. If this is the case, solve the hardware resource problem first. You may still need to follow on with a plan for multiple regions.

Information from at least three levels is required:

1. **CICS**: Examine the CICS interval or end-of-day statistics for exceptions, queues, and other symptoms which suggest overloads on specific resources. A shorter reporting period can isolate a problem. Consider software as well as hardware resources: for example, utilization of VSAM strings or database threads as well as files and TP lines. Check run time messages sent to the console and to transient data destinations, such as CSMT and CSTL, for persistent application problems and network errors.

   Use tools such as CEMT and RMF, to monitor the online system and identify activity which correlates to periods of bad performance. Collect CICS monitoring facility history and analyze it, using tools like SLR, to identify performance and resource usage exceptions and trends. For example, processor-intensive transactions which do little or no I/O should be noted. After they get control, they can monopolize the processor. This can cause erratic response in other transactions with more normally balanced activity profiles. They may be candidates for isolation in another CICS region.
2. **MVS**: Use SMF data to discover any relationships between periods of bad CICS performance and other concurrent activity in the MVS system. Use RMF data to identify overloaded devices and paths. Monitor CICS region paging

rates to make sure that there is sufficient real storage to support the configuration.

3. **Network**: The proportion of response time spent in the system is usually small compared with transmission delays and queuing in the network. Use tools such as NetView, NPM, and VTAMPARS to identify problems and overloads in the network. Without automatic tools like these, you are dependent on the application users' subjective opinions that performance has deteriorated. This makes it more difficult to know how much worse performance has become and to identify the underlying reasons.

Within CICS, the performance problem is either a poor response time or an unexpected and unexplained high use of resources. In general, you need to look at the system in some detail to see why tasks are progressing slowly through the system, or why a given resource is being used heavily. The best way of looking at detailed CICS behavior is by using CICS auxiliary trace. But note that switching on auxiliary trace, though the best approach, may actually worsen existing poor performance while it is in use (see page 306).

The approach is to get a picture of task activity first, listing only the task traces, and then to focus on particular activities: specific tasks, or a very specific time interval. For example, for a response time problem, you might want to look at the detailed traces of one task that is observed to be slow. There may be a number of possible reasons.

The tasks may simply be trying to do too much work for the system. You are asking it to do too many things, it clearly takes time, and the users are simply trying to put too much through a system that can't do all the work that they want done.

Another possibility is that the system is real-storage constrained, and therefore the tasks progress more slowly than expected because of paging interrupts. These would show as delays between successive requests recorded in the CICS trace.

Yet another possibility is that many of the CICS tasks are waiting because there is contention for a particular function. There is a wait on strings on a particular data set, for example, or there is an application enqueue such that all the tasks issue an enqueue for a particular item, and most of them have to wait while one task actually does the work. Auxiliary trace enables you to distinguish most of these cases.

Other ways to analyze performance

Potentially, any performance measurement tool, including statistics and the CICS monitoring facility, may tell you something about your system that help in diagnosing problems. You should regard each performance tool as usable in some degree for each purpose: monitoring, single-transaction measurement, and problem determination.

Again, CICS statistics may reveal heavy use of some resource. For example, you may find a very large allocation of temporary storage in main storage, a very high number of storage control requests per task (perhaps 50 or 100), or high program use counts that may imply heavy use of program control LINK.

Both statistics and CICS monitoring may show exceptional conditions arising in the CICS run. Statistics can show waits on strings, waits for VSAM shared resources, waits for storage in GETMAIN requests, and so on. These also generate CICS monitoring facility exception class records.

While these conditions are also evident in CICS auxiliary trace, they may not appear so obviously, and the other information sources are useful in directing the investigation of the trace data.

In addition, you may gain useful data from the investigation of CICS outages. If there is a series of outages, common links between the outages should be investigated.

The next chapter tells you how to identify the various forms of CICS constraints, and Chapter 10 gives you more information on performance analysis techniques.

Chapter 12. Identifying CICS constraints

If current performance has been determined to be unacceptable, you need to identify the performance constraints (that is, the causes of the symptoms) so that they can be tuned.

Major CICS constraints

Major constraints on a CICS system show themselves in the form of external symptoms: stress conditions and paging being the chief forms. This chapter describes these symptoms in some detail so that you can recognize them when your system has a performance problem, and know the ways in which CICS itself attempts to resolve various conditions.

The fundamental thing that has to be understood is that practically every symptom of poor performance arises in a system that is congested. For example, if there is a slowdown in DASD, transactions doing data set activity pile up: there are waits on strings; there are more transactions in the system, there is therefore a greater virtual storage demand; there is a greater real storage demand; there is paging; and, because there are more transactions in the system, the task dispatcher uses more processor power scanning the task chains. You then get task constraints, your MXT or transaction class limit is exceeded and adds to the processor overhead because of retries, and so on.

The result is that the system shows heavy use of ***all*** its resources, and this is the typical system stress. It does not mean that there is a problem with all of them; it means that there is a constraint that has yet to be found. To find the constraint, you have to find what is really affecting task life.

Response times

The basic criterion of performance in a production system is response time, but what is good response time? In straightforward data-entry systems, good response time implies subsecond response time. In normal production systems, good response time is measured in the five to ten second range. In scientific, compute-bound systems or in print systems, good response time can be one or two minutes.

Good performance, then, depends on a variety of factors including user requirements, available capacity, system reliability, and application design. Good performance for one system can be poor performance for another.

When checking whether the performance of a CICS system is in line with the system's expected or required capability, you should base this investigation on the hardware, software, and applications that are present in the installation.

If, for example, an application requires 100 accesses to a database, a response time of three to six seconds may be considered to be quite good. If an application requires only one access, however, a response time of three to six seconds for disk accesses would need to be investigated. Response times, however, depend on the speed of the processor, and on the nature of the application being run on the production system.

You should also observe how consistent the response times are. Sharp variations indicate erratic system behavior.

The typical way in which the response time in the system may vary with increasing transaction rate is gradual at first, then deteriorates rapidly and suddenly. The typical curve shows a sharp change when, suddenly, the response time increases dramatically for a relatively small increase in the transaction rate.

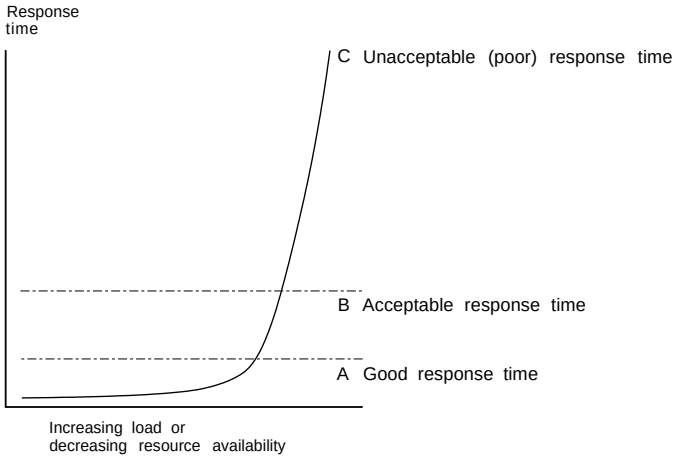


Figure 31. Graph to show the effect of response time against increasing load

For stable performance, it is necessary to keep the system operating below this point where the response time dramatically increases. In these circumstances, the user community is less likely to be seriously affected by the tuning activities being undertaken by the DP department, and these changes can be done in an unhurried and controlled manner.

Response time can be considered as being made up of queue time and service time. Service time is generally independent of usage, but queue time is not. For example, 50% usage implies a queue time approximately equal to service time, and 80% usage implies a queue time approximately four times the service time. If service time for a particular system is only a small component of the system response, for example, in the processor, 80% usage may be acceptable. If it is a greater portion of the system response time, for example, in a communication line, 50% usage may be considered high.

If you are trying to find the response time from a terminal to a terminal, you should be aware that the most common “response time” obtainable from any aid or tool that runs in the host is the “internal response time.” Trace can identify only when the software in the host, that is, CICS and its attendant software, first “sees” the message on the inbound side, and when it last “sees” the message on the outbound side.

Internal response time gives no indication of how long a message took to get from the terminal, through its control unit, across a line of whatever speed, through the communication controller (whatever it is), through the communication access method (whatever it is), and any delays before the channel program that initiated the read is finally posted to CICS. Nor does it account for the time it might take for CICS to start processing this input message. There may have been lots of work for CICS to do before terminal control regained control and before terminal control even found this posted event.

The same is true on the outbound side. CICS auxiliary trace knows when the application issued its request, but that has little to do with when terminal control found the request, when the access method ships it out, when the controllers can get to the device, and so on.

While the outward symptom of poor performance is overall bad response, there are progressive sets of early warning conditions which, if correctly interpreted, can ease the problem of locating the constraint and removing it.

In the advice given so far, we have assumed that CICS is the only major program running in your system. If batch programs or other online programs are running simultaneously with CICS, you must ensure that CICS receives its fair share of the system resources and that interference from other regions does not seriously degrade CICS performance.

Storage stress

Stress is the term used in CICS for a shortage of free space in one of the dynamic storage areas.

Storage stress can be a symptom of other resource constraints that cause CICS tasks to occupy storage for longer than is normally necessary, or of a flood of tasks which simply overwhelms available free storage, or of badly designed applications that require unreasonably large amounts of storage.

Controlling storage stress

Compared with previous versions of CICS, CICS/ESA 4.1 handles storage stress in a different way. The intention is to avoid the major disturbance to transaction throughput and response time that could result from dynamic program storage compression. (The removal of ***all*** nonresident, not-in-use programs which previous versions implemented when a GETMAIN request could not be satisfied.)

Nonresident, not-in-use programs may be deleted progressively with decreasing free storage availability as CICS determines appropriate, on a least-recently-used basis. The dispatching of new tasks is also progressively slowed as free storage approaches a critically small amount. This self-tuned activity tends to spread the cost of managing storage. There may be less or more program loading overall, but the heavy overhead of a full program compression is not incurred at the critical time.

The loading or reloading of programs is handled by CICS with an MVS subtask. This allows other user tasks to proceed if a processor of the MVS image is available and even if a page-in is required as part of the program load.

| User runtime control of storage usage is achieved through appropriate use of MXT
| and transaction class limits, (which may alter XDSALIMIT dynamically). This is necessary to avoid the short-on-storage condition that can result from unconstrained demand for storage.

Short-on-storage condition

CICS reserves a minimum number of free storage pages for use only when there is not enough free storage to satisfy an unconditional GETMAIN request even when all, not-in-use, nonresident programs have been deleted.

Whenever a request for storage results in the number of contiguous free pages in one of the dynamic storage areas falling below its respective cushion size, or failing to be satisfied even with the storage cushion, a cushion stress condition exists. Details are given in the storage manager statistics ("Times request suspended," "Times cushion released"). CICS attempts to alleviate the storage stress situation by releasing programs with no current user and slowing the attachment of new tasks. If these actions fail to alleviate the situation or if the stress condition is caused by a task that is suspended for SOS, a short-on-storage condition is signaled. This is accompanied by message DFHSM0131 or DFHSM0133.

Purging of tasks

If a CICS task is suspended for longer than its DTIMOUT value, it may be purged. That is, the task is abended and its resources freed, thus allowing other tasks to use those resources. In this way, CICS attempts to resolve what is effectively a deadlock on storage.

CICS abend

If purging tasks is not possible or not sufficient to solve the problem, CICS ceases processing. You must then either cancel and restart the CICS system, or initiate or allow an XRF takeover.

Effect of program loading on CICS

CICS/ESA 4.1 employs MVS load under an MVS subtask to load programs. This provides the benefits, relative to previous versions, of fast loading from DASD and allows the use of the library lookaside function of MVS to eliminate most DASD I/Os by keeping copies of programs in an MVS controlled dataspace exploiting expanded storage.

A page-in operation causes the MVS task which requires it to stop until the page has been retrieved. If the page is to be retrieved from DASD, this has a significant effect. When the page can be retrieved from expanded storage, the impact is only a relatively small increase in processor usage.

The loading of a program into CICS storage can be a major cause of page-ins. Because this is carried out under a subtask separate from CICS main activity, such page-ins do not halt most other CICS activities.

What is paging?

The virtual storage of a processor may far exceed the size of the central storage available in the configuration. Any excess must be maintained in auxiliary storage (DASD), or in expanded storage. This virtual storage occurs in blocks of addresses called "pages". Only the most recently referenced pages of virtual storage are assigned to occupy blocks of physical central storage. When reference is made to a page of virtual storage that does not appear in central storage, the page is

brought in from DASD or expanded storage to replace a page in central storage that is not in use and least recently used.

The newly referenced page is said to have been “paged in”. The displaced page may need to be “paged out” if it has been changed.

Paging problems

It is the **page-in** rate that is of primary concern, because page-in activity occurs synchronously (that is, an MVS task stops until the page fault is resolved). Page-out activity is overlapped with CICS processing, so it does not appreciably affect CICS throughput.

A page-in from expanded storage incurs only a small processor usage cost, but a page-in from DASD incurs a time cost for the physical I/O and a more significant increase in processor usage.

Thus, extra DASD page-in activity slows down the rate at which transactions flow through the CICS system, that is, transactions take longer to get through CICS, you get more overlap of transactions in CICS, and so you need more virtual and real storage.

If you suspect that a performance problem is related to excessive paging, you can use RMF to obtain the paging rates.

Consider controlling CICS throughput by using MXT and transaction class limits in CICS, or the DL/I CMAXTSK specification in the DLZACT, on the basis that a smaller number of concurrent transactions requires less real storage, cause less paging, and be processed faster than a larger number of transactions.

| When a CICS system is running with transaction isolation active, storage is allocated to user transactions in multiples of 1MB. This means that the virtual storage requirement for a CICS system with transaction isolation enabled is very large. This does not directly affect paging which only affects those 4K byte pages that have been touched. More real storage is required in ELSQA, however, and for more information on transaction isolation and real storage see “Transaction isolation and real storage requirements” on page 274.

What is an ideal CICS paging rate from DASD? Less than one page-in per second is best to maximize the throughput capacity of the CICS region. Anything less than five page-ins per second is probably acceptable; up to ten may be tolerable. Ten per second is marginal, more is probably a major problem. Because CICS performance can be affected by the waits associated with paging, you should not allow paging to exceed more than five to ten pages per second.

Note: The degree of sensitivity of CICS systems to paging from DASD depends on the transaction rate, the processor loading, and the average internal lifetime of the CICS tasks. An ongoing, hour-on-hour rate of even five page-faults per second may be excessive for some systems, particularly when you realize that peak paging rates over periods of ten seconds or so could easily be four times that figure.

What paging rates are excessive on various processors and are these rates operating-system dependent? Excessive paging rates should be defined as those which cause excessive delays to applications. The contribution caused by the high-priority paging supervisor executing instructions and causing applications to

wait for the processor is probably a minor consideration as far as overall delays to applications are concerned. Waiting on a DASD device is the dominant part of the overall delays. This means that the penalty of “high” paging rates has almost nothing to do with the processor type.

CICS systems are usually able to deliver much better response times with somewhat better processor utilization when the potential of large amounts of central and expanded storage is exploited by keeping more data and programs in memory.

Recovery from storage violation

CICS can detect storage violations when:

- The duplicate storage accounting area (SAA) or the initial SAA of a TIOA storage element has become corrupted.
- The leading storage check zone or the trailing storage check zone of a user task storage has become corrupted.

A storage violation can occur in two basic situations:

1. When CICS detects an error during its normal processing of a FREEMAIN request for an individual element of a TIOA storage, and finds that the two storage check zones of the duplicate SAA and the initial SAA are not identical.
2. CICS also detects user violations involving user task storage by checking the storage check zones of an element of user task storage following a FREEMAIN command.

When a storage violation is detected, an exception trace entry is made in the internal trace table. A message (DFHSM0102) is issued and a CICS system dump follows if the dump option is switched on.

Storage violations can be reduced considerably if CICS has storage protection transaction and command protection enabled.

See the *CICS/ESA Problem Determination Guide*. for further information about diagnosing and dealing with storage violations.

Dealing with limit conditions

The main limit conditions or constraints that can occur in a CICS system include those listed at the beginning of this chapter. Stress conditions generally tell you that certain limiting conditions have been reached. If these conditions occur, additional processing is required, and the transactions involved have to wait until resources are released.

To summarize, limit conditions can be indicated by the following:

- Virtual storage conditions ("short-on-storage": SOS). This item in the CICS storage manager statistics shows a deficiency in the allocation of virtual storage space to the CICS region.

  In most circumstances, allocation of more virtual storage does not in itself cause a degradation of performance. You should determine the reason for the condition in case it is caused by some form of error. This could include failure of applications to free storage (including temporary storage), unwanted multiple copies of programs or maps, storage violations, and high activity of nonresident exception routines caused by program or hardware errors.

  All new applications should be written to run above the 16MB line. The dynamic storage areas above the 16MB line can be expanded up to the 2GB limit of 31-bit addressing. The dynamic storage areas below the 16MB line are limited to less than the region size, which is less than 16MB.
- Number of simultaneous tasks (MXT and transaction class limit) reached (shown in the transaction manager statistics).
- Maximum number of VTAM receive-any RPLs in use (shown in the VTAM statistics).
- 'Wait-on-string' and associated conditions for VSAM data sets (shown in the file control statistics).

Check how frequently the limit conditions occur. In general:

- If **no** limit conditions occur, this implies that too many resources have been allocated. This is quite acceptable if the resource is inexpensive, but not if the resource is both overallocated and of more use elsewhere.
- **Infrequent** occurrence of a limit condition is an indication of good usage of the particular resource. This usually implies a healthy system.
- **Frequent** occurrence (greater than 5% of transactions) usually reveals a problem, either directly or indirectly, that needs action to prevent more obvious signs of poor performance. If the frequency is greater than about 10%, you may have to take some action quickly because the actions taken by CICS itself (dynamic program storage compression, release of storage cushion, and so on) can have a perceptible effect on performance.

  Your own actions should include:

  – Checking for errors

  – Raising the limit, provided that it does not have a degrading effect on other areas

  – Allocating more resources to remove contention

  – Checking recovery usage for contention.

Identifying performance constraints

When you are dealing with limit conditions, you may find it helpful to check the various points where performance constraints can exist in a system. These points are summarized below under hardware and software constraints.

Hardware constraints

1. **Processor cycles**. It is not uncommon for transactions to execute more than one million instructions. To execute these instructions, they must contend with other tasks and jobs in the system. At different times, these tasks must wait for such activities as file I/O. Transactions give up their use of the processor at these points and must contend for use of the processor again when the activity has completed. Dispatching priorities affect which transactions or jobs get use of the processor, and batch or other online systems may affect response time through receiving preferential access to the processor. Batch programs accessing online databases also tie up those databases for longer periods of time if their dispatching priority is low. At higher usages, the wait time for access to the processor can be significant.

2. **Real storage (working set)**. Just as transactions must contend for the processor, they also must be given a certain amount of real storage. A real storage shortage can be particularly significant in CICS performance because a normal page fault to acquire real storage results in synchronous I/O. The basic design of CICS is asynchronous, which means that CICS processes requests from multiple tasks concurrently to make maximum use of the processor. Most paging I/O is synchronous and causes the MVS task that CICS is using to wait, and that part of CICS cannot do any further processing until the page operation completes. Most, but not all, of CICS processing uses a single MVS task (called 'QUASI' in the dispatcher statistics).

3. **Database-associated hardware (I/O) contention**. When data is being accessed to provide information that is required in a transaction, an I/O operation passes through the processor, the processor channel, a disk control unit, the head of string on a string of disks, and the actual disk device where the data resides. If any of these devices are overused, the time taken to access the data can increase significantly. This overuse can be the result of activity on one data set, or on a combination of active data sets. Error rates also affect the usage and performance of the device. In shared DASD environments, contention between processors also affects performance. This, in turn, increases the time that the transaction ties up real and virtual storage and other resources.

   The use of large amounts of central and expanded storage by using very large data buffers, and by keeping programs in storage, can significantly reduce DB I/O contention and somewhat reduce processor utilization while delivering significant internal response time benefits.

4. **Network-associated hardware contention**. The input and output messages of a transaction must pass from the terminal to a control unit, a communications link, a network controller, a processor channel, and finally the processor. Just as overuse of devices to access data can affect response time, so excessive use of network resources can cause performance degradation. Error rates affect performance as well. In some cases, the delivery of the output message is a prerequisite to freeing the processor resources that are accessed, and contention can cause these resources to be tied up for longer periods.

Software constraints

1. **Database design**. A data set or database needs to be designed to the needs of the application it is supporting. Such factors as the pattern of access to the data set (especially whether it is random or sequential), access methods chosen, and the frequency of access determine the best database design. Such data set characteristics as physical record size, blocking factors, the use of alternate or secondary indexes, the hierarchical or relational structure of database segments, database organization (HDAM, HIDAM, and so on), and pointer arrangements are all factors in database performance.

   The length of time between data set reorganizations can also affect performance. The efficiency of accesses decreases as the data set becomes more and more fragmented. This fragmentation can be kept to the minimum by reducing the length of time between data set reorganizations.

2. **Network design**. This item can often be a major factor in response time because the network links are much slower than most components of an online system. Processor operations are measured in nanoseconds, line speeds in seconds. Screen design can also have a significant effect on overall response time. A 1200-byte message takes one second to be transmitted on a relatively high-speed 9600 bits-per-second link. If 600 bytes of the message are not needed, half a second of response time is wasted. Besides screen design and size, such factors as how many terminals are on a line, the protocols used (SNA, bisynchronous), and full-or half-duplex capabilities can affect performance.

3. **Use of specific software interfaces or serial functions**. The operating system, terminal access method, database manager, data set access method, and CICS must all communicate in the processing of a transaction. Only a given level of concurrent processing can occur at these points, and this can also cause a performance constraint. Examples of this include the VTAM receive any pool (RAPOOL), VSAM data set access (strings), CICS temporary storage, CICS transient data, and CICS intercommunication sessions. Each of these can have a single or multiserver queueing effect on a transaction's response time, and can tie up other resources by slowing task throughput.

One useful technique for isolating a performance constraint in a CICS system with VTAM is to use the IBMTEST command issued from a user's terminal. This terminal must not be in session with CICS, but must be connected to VTAM.

You enter at a VTAM terminal:

```
IBMTEST (n)(,data)
```

where *n* is the number of times you want the data echoed, and *data* may consist of any character string. If you enter no data, the alphabet and the numbers zero through nine are returned to the terminal. This command is responded to by VTAM.

IBMTEST is an echo test designed to give the user a rough idea of the VTAM component of terminal response time. If the response time is fast in a slow-response system, the constraint is not likely to be any component from VTAM onward. If this response is slow, VTAM or the network may be the reason. This sort of deductive process in general can be useful in isolating constraints.

To avoid going into session with CICS, you may have to remove APPLID= from the LU statement or CONNECT=AUTO from the TCT TYPE=TERMINAL macro.

Resource contention

The major resources used or managed by CICS consist of the following:

- Processor
- Real storage
- Virtual storage
- Software (specification limits)
- Channels
- Control units
- Lines
- Devices
- Sessions to connected CICS systems.

Contention at lower levels prevents full use of higher-level resources. To avoid or reduce resource contention, you can:

- Minimize or eliminate the use of a resource by:
  - Reordering, relocating, or reducing its size
  - Redesign, rewriting, rescheduling, or reducing processing time
  - Education, eliminating a function, or controlling its usage.
- Give the resource more capacity
- Exchange one resource with another:
  - Processor with virtual storage
  - Real storage with paging I/O
  - Paging I/O with program library I/O
  - Priorities of various end-users with each other
  - CICS response times with batch throughput
  - Batch throughput with more DP operators.

Two sets of “Symptoms and Solutions” are provided in this chapter. The first set provides suggested solutions for poor response, and the second set provides suggested solutions for a variety of resource contention problems.

Solutions for poor response time

Table 8 shows four levels of response time, in decreasing order of severity. The major causes are shown for each level together with a range of suggested solutions. Your first step is to check the causes by following the advice given in Chapter 13, “ CICS performance analysis” on page 159. When you have identified the precise causes, the relevant checklist in Chapter 15, “ Performance checklists” on page 173 tells you what solutions are available and where to find information in Part 4 of this book on how to implement the solutions.

Table 8. CICS response time checklist

Level	Symptom	Major Causes	Overall Solution
1	Poor response at all loads for all transactions	High level of paging	Reduce working set, or allocate more real storage
		Very high usage of major resources	Reconsider system resource requirements and redesign system Check for application errors and resource contention
2	Poor response at medium and high loads	High level of paging	Reduce working set, or allocate more real storage
		High processor usage	Reduce pathlength, or increase processor power
		High DB or data set usage	Reorganize data sets, or reduce data transfer, or increase capacity
		High communication network usage	Reduce data transfer, or increase capacity
		TP or I/O access-method constraint	Increase buffer availability
		CICS limit values exceeded	Change operands, or provide more resources, or check if errors in application
3	Poor response for certain transactions only	Identify common characteristics	As for level 2
		Lines or terminal usage	Increase capacity, or reduce data transfer, or change transaction logic or data set design
		Data set usage	Change data set placement buffer allocations or change enqueue logic or data set design
		High storage usage	Redesign or tune applications
		Same subprograms as affected transaction	Redesign or tune application subprograms
		Same access method or CICS features as used by transaction	Reallocate resource or change application. Reevaluate use of feature in question
		Limit conditions	Reallocate resource or change application
4	Poor response for certain terminals	Check network loading as appropriate	Increase capacity of that part of network
		Check operator techniques	Revise terminal procedures
		Check CEDA or TCT generation	Redefine CEDA or TCT entries

Symptoms and solutions for resource contention problems

This section presents a general range of solutions for each type of constraint. You should:

1. Confirm that your diagnosis of the type of constraint is correct, by means of detailed performance analysis. Chapter 13, " CICS performance analysis" on page 159 describes various techniques.
2. Read Chapter 14, " Tuning the system" on page 167 for general advice on performance tuning.
3. See the relevant sections in Part 4 of this book for detailed information on applying the various solutions.
4. Improve virtual storage exploitation. This requires:
   - Large data buffers above the 16MB line or in Hiperspace
   - Programs that run above the 16MB line
   - Large amounts of central and expanded storage to support the virtual storage exploitation.

   Such a system can deliver better internal response times, while minimizing DASD I/O constraint and reducing processor utilization.

DASD constraint

Symptoms

- Slow response times (the length of the response time depends on the number of I/O operations, with a longer response time when batch mode is active)
- High DSA utilization
- High paging rates
- MXT limit frequently reached
- SOS condition often occurs.

Solutions

- Reduce the number of I/O operations.
- Tune the remaining I/O operations.
- Balance the I/O operations load.

See "DASD tuning" on page 191 for suggested solutions.

Communications network constraint

Symptoms

- Slow response times
- Good response when few terminals are active on a line, but poor response when many terminals are active on that line
- Big difference between internal response time and terminal response time.

Solutions

- Reduce the line utilization.
- Reduce delays in data transmission.
- Alter the network.

Remote systems constraints

Symptoms

- SOS condition or MXT occur when there is a problem with a connected region.
- CICS takes time to recover when the problem is fixed.

Solutions

- Control the amount of queuing which takes place for the use of the connections to the remote systems.
- Improve the response time of the remote system.

Virtual storage constraint

Symptoms

- Slow response times
- Multiple loads of the same program
- Increased I/O operations against program libraries
- High paging rates
- SOS condition often occurs.

Solutions

- Tune the MVS system to obtain more virtual storage for CICS (increase the region size).
- Expand or make more efficient use of the dynamic storage area.

See the “Virtual storage above and below 16MB line checklist” on page 175 for a detailed list of suggested solutions.

Real storage constraint

Symptoms

- High paging rates
- Slow response times
- MXT limit frequently reached
- SOS condition often occurs.

Solutions

- Reduce the demands on real storage
- Tune the MVS system to obtain more real storage for CICS
- Obtain more central and expanded storage.

See the “Real storage checklist” on page 176 for a detailed list of suggested solutions.

Processor cycles constraint

Symptoms

- Slow response times
- Low-priority transactions respond very slowly
- Low-priority work gets done very slowly.

Solutions

- Increase the dispatching priority of CICS.
- Reevaluate the relative priorities of operating system jobs.
- Reduce the number of MVS regions (batch).
- Reduce the processor utilization for productive work.
- Use only the CICS facilities that you really require.
- Turn off any trace that is not being used.
- Minimize the data being traced by reducing the:
  - Scope of the trace
  - Frequency of running trace.
- Obtain a faster processor.

See the “Processor cycles checklist” on page 177 for a detailed list of suggested solutions.

Chapter 13. CICS performance analysis

Performance analysis, as compared with monitoring, is the use of certain performance tools described in Part 2 to:

- Investigate a deviation from performance objectives that is resulting in performance deterioration, and identify performance problems
- Identify where a system can be adjusted to give a required level of performance
- Characterize and calibrate individual stand-alone transactions as part of the documentation of those transactions, and for comparison with some future time when, perhaps, they start behaving differently.

Assessing the performance of a DB/DC system

You may find the following performance measurements helpful in determining the performance of a system:

1. **Processor usage**: This item reflects how active the processor is. Although the central processor is of primary concern, 37X5 communications controllers and terminal control units (these can include an intelligent cluster controller such as the 3601 and also the 3270 cluster control units) can also increase response time if they are heavily used.
2. **I/O rates**: These rates measure the amount of access to a disk device or data set over a given period of time. Again, acceptable rates vary depending on the speed of the hardware and response time requirements.
3. **Terminal message or data set record block sizes**: These factors, when combined with I/O rates, provide information on the current load on the network or DASD subsystem.
4. **Indications of internal virtual storage limits**: These vary by software component, including storage or buffer expansion counts, system messages, and program abends because of system stalls. In CICS, program fetches on nonresident programs and system short-on-storage or stress messages reflect this condition.
5. **Paging rates**: CICS can be sensitive to a real storage shortage, and paging rates reflect this shortage. Acceptable paging to DASD rates vary with the speed of the DASD and response time criteria. Paging rates to expanded storage are only as important as its effect on processor usage.
6. **Error rates**: Errors can occur at any point in an online system. If the errors are recoverable, they can go unnoticed, but they put an additional load on the resource on which they are occurring.

You should investigate both system conditions and application conditions.

System conditions

A knowledge of these conditions enables you evaluate the performance of the system as a whole:

- System transaction rate (average and peak)
- Internal response time and terminal response time, preferably compared with transaction rate
- Working set, at average and peak transaction rates
- Average number of disk accesses per unit time (total, per channel, and per device)
- Processor usage, compared with transaction rate
- Number of page faults per second, compared with transaction rate and real storage
- Communication line usage (net and actual)
- Average number of active CICS tasks
- Number and duration of outages.

Application conditions

These conditions, measured both for individual transaction types and for the total system, give you an estimate of the behavior of individual application programs.

You should gather data for each main transaction and average values for the total system. This data includes:

- Program calls per transaction
- CICS storage GETMAINs and FREEMAINs (number and amount)
- Application program and transaction usage
- File control (data set, type of request)
- Terminal control (terminal, number of inputs and outputs)
- Other CICS requests.

Methods of performance analysis

You can use two methods for performance analysis:

1. Measuring a system under full production load (**full-load** measurement), to get all information that is measurable only under high system-loading.
2. Measuring single-application transactions (**single-transaction** measurement), during which the system should not carry out any other activities. This gives an insight into the behavior of single transactions under optimum system conditions.

Because a system can have a variety of problems, we cannot recommend which option you should use to investigate the behavior of a system. When in doubt about the extent of a problem, you should always use both methods.

Rapid performance degradation often occurs after a threshold is exceeded and the system approaches its ultimate load. You can see various indications only when the system is fully loaded (for example, paging, short-on-storage condition in CICS, and so on), and you should usually plan for a full-load measurement.

Bear in mind that the performance constraints might possibly vary at different times of the day. You might want to run a particular option that puts a particular pressure on the system only at a certain time in the afternoon.

If a full-load measurement reveals no serious problems, or if a system is not reaching its expected performance capability under normal operating conditions, you can then use single-transaction measurement to reveal how individual system transactions behave and to identify the areas for possible improvement.

Often, because you have no reliable information at the beginning of an investigation into the probable causes of performance problems, you have to examine and analyze the whole system.

Before carrying out this analysis, you must have a clear picture of the functions and the interactions of the following components:

- Operating system supervisor with the appropriate access methods
- CICS management modules and control tables
- VSAM data sets
- DL/I databases
- DB2
- External security managers
- Performance monitors
- CICS application programs
- Influence of other regions
- Hardware peripherals (disks and tapes).

In addition, you should collect the following information:

- Does performance fluctuate or is it uniformly bad?
- Are performance problems related to a specific hour, day, week, or month?
- Has anything in the system been changed recently?
- Have all such changes been fully documented?

Full-load measurement

A full-load measurement highlights latent problems in the system. It is important that full-load measurement lives up to its name, that is, you should make the measurement when, from production experience, the peak load is reached. Many installations have a peak load for about one hour in the morning and again in the afternoon. CICS statistics and various performance tools can provide valuable information for full-load measurement. In addition to the overall results of these tools, it may be useful to have the CICS auxiliary trace or RMF active for about one minute.

CICS auxiliary trace

CICS auxiliary trace can be used to find situations that occur under full load. For example, all ENQUEUEs that cannot immediately be honored in application programs result in a suspension of the issuing task. If this frequently happens, attempts to control the system by using the CEMT master transaction, are not effective.

Trace is a very heavy overhead. Use trace selectivity options to minimize this overhead.

RMF

It is advisable to do the RMF measurement without any batch activity. (See “Resource measurement facility (RMF)” on page 32 for a detailed description of this tool. Guidance on how to use RMF with the CICS monitoring facility is given in “Using CICS monitoring SYSEVENT information with RMF” on page 67.)

For full-load measurement, the system activity report and the DASD activity report are important.

The most important values for full-load measurement are:

- Processor usage
- Channel and disk usage
- Disk unit usage
- Overlapping of processor with channel and disk activity
- Paging
- Count of start I/O operations and average start I/O time
- Response times
- Transaction rates.

You should expect stagnant throughput and sharply climbing response times as the processor load approaches 100%.

It is difficult to forecast the system paging rate that can be achieved without serious detriment to performance, because too many factors interact. You should observe the reported paging rates; note that short-duration severe paging leads to a rapid increase in response times.

In addition to taking note of the count of start I/O operations and their average length, you should also find out whether the system is waiting on one device only. With disks, for example, it can happen that several frequently accessed data sets are on one disk and the accesses interfere with each other. In each case, you should investigate whether a system wait on a particular unit could not be minimized by reorganizing the data sets.

The RMF DASD activity report includes the following information:

- A summary of all disk information
- Per disk, a breakdown by system number and region
- Per disk, the distribution of the seek arm movements
- Per disk, the distribution of accesses with and without arm movement.

| Use IOQ(DASD) option in RMF monitor 1 to show DASD control unit contention.

After checking the relationship of accesses with and without arm movement, for example, you may want to move to separate disks those data sets that are periodically very frequently accessed.

Comparison charts

You might wish to consider using a comparison chart to measure key aspects of your system's performance before and after tuning changes have been made. A suggested chart is as follows:

Table 9. Comparison chart 1

Run	DL/I transactions	VSAM transactions	Response times	Most heavily used transaction	Average-use transaction
	No./response	No./response	DL/I VSAM	No./response	No./response

Table 10. Comparison chart 2

Run	Average-use transaction	Paging rate	DSA virtual storage	Tasks	Most heavily used DASD
	No./response	System/CICS	Max./Average	Peak/at MXT	Resp./Util

Table 11. Comparison chart 3

Run	Average-use DASD	CPU utilization			
	Resp./Util				

The use of this type of comparison chart requires the use of TPNS, RMF, and CICS interval statistics running together for about 20 minutes, at a peak time for your system. It also requires you to identify the following:

- A representative selection of terminal-oriented DL/I transactions accessing DL/I databases
- A representative selection of terminal-oriented transactions processing VSAM files
- The most heavily used transaction
- Two average-use nonterminal-oriented transactions writing data to intrapartition transient data destinations
- The most heavily used volume in your system
- A representative average-use volume in your system.

To complete the comparison chart for each CICS run before and after a tuning change, you can obtain the figures from the following sources:

- **DL/I transactions**: you should first identify a selection of terminal-oriented DL/I transactions accessing DL/I databases.
- **VSAM transactions**: similarly, you should first identify a selection of terminal-oriented transactions processing VSAM files.
- **Response times**: external response times are available from the TPNS terminal response time analysis report; internal response times are available from RMF. The “DL/I” subheading is the average response time calculated at the 99th percentile for the terminal-oriented DL/I transactions you have previously selected. The “VSAM” subheading is the average response time calculated at the 99th percentile for the terminal-oriented VSAM transactions you have previously selected.
- **Paging rate (system)**: this is from the RMF paging activity report, and is the figure shown for total system non-VIO non-swap page-ins added to the figure shown for the total system non-VIO non-swap page-outs. This is the total paging rate per second for the entire system.
- **Tasks**: this is from the transaction manager statistics (part of the CICS interval, end-of-day, and requested statistics). The “Peak” subheading is the figure shown for “Peak Number of Tasks” in the statistics. The “At MXT” subheading is the figure shown for “Number of Times at Max. Task” in the statistics.
- **Most heavily used DASD**: this is from the RMF direct access device activity report, and relates to the most heavily used volume in your system. The “Resp.” subheading is the figure shown in the “Avg. Resp. Time” column for the volume you have selected. The “Util.” subheading is the figure shown in the “% Dev. Util.” column for that volume.
- **Average-use DASD**: this is also from the RMF direct access device activity report, and relates to a representative average-use volume in your system. The “Resp.” subheading is the figure shown in the “Avg. Resp. Time” column for the volume you have selected. The “Util.” subheading is the figure shown in the “% Dev. Util.” column for that volume.
- **Processor utilization**: this is from the RMF processor activity report.

This chart is most useful when comparing before-and-after changes in performance while you are tuning your CICS system.

Single-transaction measurement

You can use full-load measurement to evaluate the average loading of the system per transaction. However, this type of measurement cannot provide you with information on the behavior of a single transaction and its possible excessive loading of the system. If, for example, nine different transaction types issue five start I/Os (SIOs) each, but the tenth issues 55 SIOs, this results in an average of ten SIOs per transaction type. This should not cause concern if they are executed simultaneously. However, an increase of the transaction rate of the tenth transaction type could possibly lead to poor performance overall.

Sometimes, response times are quite good with existing terminals, but adding a few more terminals leads to unacceptable degradation of performance. In this case, the

performance problem may be present with the existing terminals, and has simply been highlighted by the additional load.

To investigate this type of problem, do a full-load measurement as well as a single-transaction measurement. To be of any use, the single-transaction measurement must be done when no batch region is running, and there must be no activity in CICS apart from the test screen. Even the polling of remote terminals should be halted.

You should measure each existing transaction that is used in a production system or in a final test system. Test each transaction two or three times with different data values, to exclude an especially unfavorable combination of data. Document the sequence of transactions and the values entered for each test as a prerequisite for subsequent analysis or interpretation.

Between the tests of each single transaction, there should be a pause of several seconds, to make the trace easier to read. A copy of the production database or data set should be used for the test, because a test data set containing 100 records can very often result in completely different behavior when compared with a production data set containing 100 000 records.

The condition of data sets has often been the main reason for performance degradation, especially when many segments or records have been added to a database or data set. Do not do the measurements directly after a reorganization, because the database or data set is only in this condition for a short time. On the other hand, if the measurement reveals an unusually large number of disk accesses, you should reorganize the data and do a further measurement to evaluate the effect of the data reorganization.

You may feel that single-transaction measurement under these conditions with only one terminal is not an efficient tool for revealing a performance degradation that might occur when, perhaps 40 or 50 terminals are in use. Practical experience has shown, however, that this is usually the only means for revealing and rectifying, with justifiable expense, performance degradation under full load. The main reason for this is that it is sometimes a single transaction that throws the system behavior out of balance. Single-transaction measurement can be used to detect this.

Ideally, single-transaction measurement should be carried out during the final test phase of the transactions. This gives the following advantages:

- Any errors in the behavior of transactions may be revealed before production starts, and these can be put right during validation, without loading the production system unnecessarily.
- The application is documented during the measurement phase. This helps to identify the effects of later changes.

CICS auxiliary trace

Auxiliary trace is a standard feature of CICS, and gives an overview of transaction flows so that you can quickly and effectively analyze them.

From this trace, you can find out whether a specified application is running as it is expected to run. In many cases, it may be necessary for the application programmer responsible to be called in for the analysis, to explain what the transaction should actually be doing.

If you have a very large number of transactions to analyze, you can select, in a first pass, the transactions whose behavior does not comply with what is expected.

If all transactions last much longer than expected, this almost always indicates a system-wide error in application programming or in system implementation. The analysis of a few transactions is then sufficient to determine the error.

If, on the other hand, only a few transactions remain in this category, these transactions should be analyzed next, because it is highly probable that most performance problems to date arise from these.

Chapter 14. Tuning the system

When you have identified specific constraints, you will have identified the system resources that need to be tuned. The three major steps in tuning a system are:

1. Determine acceptable tuning trade-offs
2. Make the change to the system
3. Review the results of tuning.

Determining acceptable tuning trade-offs

The art of tuning can be summarized as finding and removing constraints. In most systems, the performance is limited by a single constraint. However, removing that constraint, while improving performance, inevitably reveals a different constraint, and you might often have to remove a series of constraints. Because tuning generally involves decreasing the load on one resource at the expense of increasing the load on a different resource, relieving one constraint always creates another.

A system is always constrained. You do not simply remove a constraint; you can only choose the most satisfactory constraint. Consider which resources can accept an additional load in the system without themselves becoming worse constraints.

Tuning usually involves a variety of actions that can be taken, each with its own trade-off. For example, if you have determined virtual storage to be a constraint, your tuning options may include reducing buffer allocations for data sets, or reducing terminal scan delay (ICVTSD) to shorten the task life in the processor.

The first option increases data set I/O activity, and the second option increases processor usage. If one or more of these resources are also constrained, tuning could actually cause a performance degradation by causing the other resource to be a greater constraint than the present constraint on virtual storage.

Making the change to the system

The next step in the tuning process is to make the actual system modifications that are intended to improve performance. You should consider several points when adjusting the system:

- Tuning is the technique of making small changes to the system's resource allocation and availability to achieve relatively large improvements in response time.
- Tuning is not always effective. If the system response is too long and all the system resources are lightly used, you see very little change in the CICS response times. (This is also true if the wrong resources are tuned.) In addition, if the constraint resource, for example, line capacity, is being fully used, the only solution is to provide more capacity or redesign the application (to transmit less data, in the case of line capacity).
- Do not tune just for the sake of tuning. Tune to relieve identified constraints. If you tune resources that are not the primary cause of performance problems, this has little or no effect on response time until you have relieved the major constraints, and it may actually make subsequent tuning work more difficult. If

there is any significant improvement potential, it lies in improving the performance of the resources that ***are*** major factors in the response time.

- In general, tune major constraints first, particularly those that have a significant effect on response time. Arrange the tuning actions so that items having the greatest effect are done first. In many cases, one tuning change can solve the performance problem if it addresses the cause of the degradation. Other actions may then be unnecessary. Further, improving performance in a major way can alleviate many user complaints and allow you to work in a more thorough way. The 80/20 rule applies here; a small number of system changes normally improves response time by most of the amount by which it can be improved, assuming that those changes address the main causes of performance problems.
- Make one tuning change at a time. If two changes are made at the same time, their effects may work in opposite directions and it may be difficult to tell which of them had a significant effect.
- Change allocations or definitions gradually. For example, when reducing the number of resident programs in a system, do not change all programs in a system from RES=YES to RES=NO at once. This could cause an unexpected lengthening of response times by increasing storage usage because of fragmentation, and increasing processor usage because of higher program loading activity. If you change a few programs at a time, starting with the lesser-used programs, this can give you a better idea of the overall results.

  The same rule holds true for buffer and string settings and other data set operands, transaction and program operands, and all resources where the operand can be specified individually for each resource. For the same reason, do not make large increases or decreases in the values assigned to task limits such as MXT.
- Continue to monitor constraints during the tuning process. Because each adjustment changes the constraints in a system, these constraints vary over time. If the constraint changes, tuning must be done on the new constraint because the old one is no longer the restricting influence on performance. In addition, constraints may vary at different times during the day.
- Put fallback procedures in place before starting the tuning process. As noted earlier, some tuning can cause unexpected performance results. If this leads to poorer performance, it should be reversed and something else tried. If previous definitions or path designs were not saved, they have to be redefined to put the system back the way it was, and the system continues to perform at a poorer level until these restorations are made. If the former setup is saved in such a way that it can be recalled, back out of the incorrect change becomes much simpler.

APAR

MJO 28/8/98

id=dfht3pq.Language Environment

Language Environment (LE) conforming CICS applications issuing EXEC CICS LINK requests cause an increase in system pathlength. Repeated EXEC CICS LINK calls to the same LE-conforming program result in multiple GETMAIN/FREEMAIN requests for run-time work areas (RUWAs).

RUWAPOOL(YES) results in the creation of a run-unit work area pool during task initialization. This pool is used to allocate RUWAs required by LE-conforming programs. This reduces the number of GETMAINS and FREEMAINS in tasks which perform many EXEC CICS LINKS to LE-conforming programs.

For more information, about the RUWAPOOL system initialization parameter, see the *CICS/ESA System Definition Guide*.

Reviewing the results of tuning

After each adjustment has been done, review the performance measurements that have been identified as factors of response time to verify that the desired performance changes have occurred and to quantify that change. If performance has improved to the point that service level agreements are being met, no more tuning is required. If performance is better, but not yet acceptable, investigation is required to determine the next action to be taken, and to verify that the resource that was tuned is still a constraint. If it is not still a constraint, new constraints need to be identified and tuned. This is a return to the first step of the tuning process, and you should repeat the next steps in that process until an acceptable performance level is reached.

Part 4. Improving the performance of a CICS system

Important

Always tune DASD, the network, and the overall MVS system ***before*** tuning any individual CICS subsystem through CICS parameters.

Also review your application code before any further tuning

Chapter 15, " Performance checklists" on page 173 itemizes the actions you can take to tune the performance of an operational CICS system.

The other chapters in this part contain the relevant performance tuning guidelines for the following aspects of CICS:

- Chapter 16, " MVS/ESA and DASD" on page 179
- Chapter 17, " Networking and VTAM" on page 193
- Chapter 18, " VSAM and file control" on page 213
- Chapter 19, " Database management" on page 235
- Chapter 20, " Journaling" on page 249
- Chapter 21, "Virtual and real storage" on page 255
- "Size of control intervals" on page 217
- "VSAM buffer allocations for LSR (BUFFERS)" on page 224
- "VSAM string settings for NSR (STRNO)" on page 225
- Chapter 25, " Tuning XRF" on page 311
- Chapter 26, "Improving CICS startup and normal shutdown time" on page 323.

Chapter 15. Performance checklists

The following checklists provide a quick reference to options that you can adjust to relieve different constraints. They assume that you have identified the exact cause of an existing constraint; they should ***not*** be used for random tuning exercises.

There are four checklists, corresponding to four of the main contention areas described in Chapter 12, “Identifying CICS constraints” on page 145.

1. I/O contention (this applies to data set and database subsystems, as well as to the data communications network)
2. Virtual storage above and below the 16MB line
3. Real storage
4. Processor cycles.

The checklists are in the sequence of low-level to high-level resources, and the items are ordered from those that probably have the greatest effect on performance to those that have a lesser effect, from the highest likelihood of being a factor in a normal system to the lowest, and from the easiest to the most difficult to implement.

Before taking action on a particular item, you should review the item to:

- Determine whether the item is applicable in your particular environment
- Understand the nature of the change
- Identify the trade-offs involved in the change.

Input/output contention checklist

Note:

Ideally, I/O contention should be reduced by using very large data buffers and keeping programs in storage. This would require adequate central and expanded storage, and programs that can be loaded above the 16MB line

Item	Page
VSAM considerations	
Review use of LLA	189
Implement Hiperspace buffers	228
Review/increase data set buffer allocations within LSR	229
Use data tables when appropriate	231
Database considerations	
Replace shared database access and DL/I function shipping with IMS/ESA DBCTL facility	241
Reduce/replace shared database access to online data sets	241
Increase DL/I threads and buffers	238
Increase DL/I storage pools	239
Review DB2 threads and buffers	243
Journaling	
Increase activity keypoint frequency (AKPFREQ) value	249
Adjust BUFSIZE appropriately	250
Optimize system log swaps	252
Terminals, VTAM and SNA.	
Implement terminal output compression exit	207
Increase concurrent VTAM inputs	196
Increase concurrent VTAM logon/logoffs	203
Minimize SNA terminal data flows	200
Reduce SNA chaining	201
Miscellaneous	
Reduce DFHRPL library contention	272
Review temporary storage strings	295
Review transient data strings	300

Virtual storage above and below 16MB line checklist

Note:

The lower the number of concurrent transactions in the system, the lower the usage of virtual storage. Therefore, improving transaction internal response time decreases virtual storage usage. Keeping programs in storage above the 16MB line, and minimizing physical I/Os makes the largest contribution to well-designed transaction internal response time improvement.

Real storage checklist

Note:

Adequate central and expanded storage is vital to achieving good performance with CICS/ESA.

Processor cycles checklist

Note:

Minimizing physical I/Os by employing large data buffers and keeping programs in storage reduces processor use, if adequate central and expanded storage is available.

Chapter 16. MVS/ESA and DASD

This chapter includes the following topics:

Tuning CICS and MVS/ESA

Tuning CICS for virtual storage under MVS/ESA depends on the following main elements:

- MVS/ESA systems tuning
- VTAM tuning
- CICS tuning
- VSAM tuning.

Because tuning is a top-down activity, you should already have made a vigorous effort to tune MVS/ESA before tuning CICS. Your main effort to reduce virtual storage constraint and to get relief should be concentrated on reducing the life of the various individual transactions: in other words, shortening task life.

This section describes some of the techniques that can contribute significantly to shorter task life, and therefore, a reduction in virtual storage constraint.

The installation of a faster processor can cause the current instructions to be executed faster and, therefore, reduce task life (internal response time), because more transactions can be processed in the same period of time. Installing faster DASD can reduce the time spent waiting for I/O completion, and this shorter wait time for paging operations, data set index retrieval, or data set buffer retrieval can also reduce task life in the processor.

Additional real storage, if page-ins are frequently occurring (if there are more than 5 to 10 page-ins per second, CICS performance is affected), can reduce waits for the paging subsystem.

MVS/ESA provides storage isolation for an MVS/ESA performance group, which allows you to reserve a specific range of real storage for the CICS address space and to control the page-rates for that address space based on the task control block (TCB) time absorbed by the CICS address space during execution.

You can isolate CICS data on DASD drives, strings, and channels to minimize the I/O contention suffered by CICS from other DASD activity in the system. Few CICS online systems generate enough I/O activity to affect the performance of CICS seriously if DASD is isolated in this manner.

So far (except when describing storage isolation and DASD sharing), we have concentrated on CICS systems that run a stand-alone single CICS address space. The sizes of all MVS/ESA address spaces are defined by the common requirements of the largest subsystem. If you want to combine the workload from two or more processors onto an MVS image, you must be aware of the virtual storage requirements of each of the subsystems that are to execute on the single-image ESA processor. Review the virtual storage effects of combining the following kinds of workload on a single-image MVS/ESA system:

1. CICS and a large number (100 or more) of TSO users
2. CICS and a large IMS system
3. CICS and 5000 to 7500 VTAM LUs.

By its nature, CICS requires a large private region that may not be available when the large system's common requirements of these other subsystems are satisfied. If, after tuning the operating system, VTAM, VSAM, and CICS, you find that your address space requirements still exceed that available, you can split CICS using one of three options:

1. Multiregion option (MRO)
2. Intersystem communication (ISC)
3. Multiple independent address spaces.

Adding large new applications or making major increases in the size of your VTAM network places large demands on virtual storage, and you must analyze them before implementing them in a production system. Careful analysis and system specification can avoid performance problems arising from the addition of new applications in a virtual-storage-constrained environment.

If you have not made the necessary preparations, you usually become aware of problems associated with severe stress only after you have attempted to implement the large application or major change in your production system. Some of these symptoms are:

- Poor response times
- Short-on-storage
- Program compression
- Heavy paging activity
- Many well-tested applications suddenly abending with new symptoms
- S80A and S40D abends
- S822 abends
- Dramatic increase in I/O activity on DFHRPL program libraries.

Various chapters in the rest of this book deal with specific, individual operands and techniques to overcome these problems. They tell you how to minimize the use of virtual storage in the CICS address space, and how to split it into multiple address spaces if your situation requires it.

For an overall description of ESA virtual storage, see Appendix C, "MVS and CICS virtual storage" on page 493.

Reducing MVS common system area requirements

This can be the most productive area for tuning. CICS installations that have not previously tuned their ESA system may be able to recover 1.5 to 2.0 megabytes of virtual storage. This topic is outside the scope of this book, but you should investigate it fully before tuning CICS. A manual that gives information on this is *MVS/ESA Initialization and Tuning Guide*.

Reviewing MVS storage

CSA is allocated on MB boundaries. Allocation can be as much as 2K over the MB boundary which causes 1MB to be lost from below the 16MB line. If your CICS system requires a larger amount of storage the below the 16MB line constraint should be removed.

Splitting online systems: availability

Splitting the CICS system into two or more separate address spaces may lead to improved availability. If CICS failures are being caused by application program errors, for example, separating out the failing application can improve overall availability. This can also give virtual storage gains and, in addition, can allow you to use multiprocessors and MVS images more efficiently. See “Splitting online systems: virtual storage” on page 256 for more details.

The availability of the overall system may be improved by splitting the system because the effects of a failure can be limited or the time to recover from the failure can be reduced.

The main ways of splitting a system for availability are to have:

- **Terminal owning regions**. With one or more terminal owning regions (TORs) using transaction routing, availability can be improved because a TOR is less likely to fail because it contains no application code. The time taken to restart the failed part of the system is reduced because the terminal sessions are maintained at failure if the TOR continues to operate.
- **Multiple application owning regions**. Using multiple application owning regions (AORs), you can separate unstable or new applications from the rest of the system. If these applications cause a failure of that AOR, all other AORs are still available. If the region susceptible to failure contains no terminals or files and databases, it also tends to restart quickly.

  Applications under test in AORs can use function shipping to access ‘live’ data, which adds to the realism of the test environment.
- **File owning regions**. File requests from many CICS regions can be function-shipped to file owning regions (FORs). The FORs contain no application code and so are unlikely to fail, so that access to files can be maintained even if other regions fail. Removing the files and databases from these other regions speeds up their recovery by removing file allocation and opening time.

Having only one FOR in a system, or logical subset of a system, can reduce the operational difficulties of restarting a system.

It is possible to split the regions in different ways to those described so far, by having many regions all of which own some terminals, some applications, and some files and databases. This type of splitting is very complex to maintain and operate, and also needs careful monitoring to ensure that the performance of the overall system is optimal. For these reasons, a structured approach with each of the regions having a clearly defined set of one type of resource is recommended.

Limitations

Splitting a CICS system requires increased real storage, increased processor cycles, and extensive planning. These overheads are described in more detail in "Splitting online systems: virtual storage" on page 256.

Recommendations

If availability of your system is an important requirement, both splitting systems and the use of XRF should be considered. The use of XRF can complement the splitting of systems by automating the recovery of the components.

When splitting your system, you should try to separate the sources of failure so that as much of the rest of the system as possible is protected against their failure, and remains available for use. Critical components should be backed up, or configured so that service can be restored with minimum delay. Since the advantages of splitting regions for availability can be compromised if the queueing of requests for remote regions is not controlled, you should also review "Intersystems Session Queue Management" on page 281.

Making CICS non-swappable

You can take a variety of actions to cause the operating system to give CICS preferential treatment in allocation of processor resources.

Making CICS non-swappable prevents the address space from being swapped out in MVS, and reduces the paging overhead. Consider leaving only very lightly used test systems swappable.

How implemented

You should consider making your CICS region non-swappable by using the PPTNSWP option in the MVS Program Properties Table (PPT).

Limitations using the PPT will make all CICS systems (including test systems) non-swappable. As an alternative, use the IPS. For more information about defining entries in the PPT see the *MVS/ESA Initialization and Tuning Guide*.

How monitored

The DISPLAY ACTIVE (DA) command on SDSF gives you an indication of the number of real pages used and the paging rate. Use RMF, the RMFMON command on TSO to provide additional information. For more information about RMF see "Resource measurement facility (RMF)" on page 32 or the *MVS/ESA RMF User's Guide*.

Isolating (fencing) real storage for CICS (PWSS and PPGRTR)

Real storage isolation, or “fencing” in MVS, is a way of allocating real storage to CICS alone, and can reduce paging problems.

Recommendations

Use PWSS=(a,b) and PPGRTR=(c,d) or PPGRT=(c,d) in the IEAIPSxx.

The PWSS=(a,b) parameter specifies the range (minimum,maximum) of page frames needed for the target working set size for an address space.

The target working set size of an XRF alternate CICS system can vary significantly in different environments.

The PPGRTR=(c,d) or PPGRT=(c,d) parameter specifies the minimum and maximum paging rates to use in adjusting the target working set size specified in PWSS. PPGRTR means that the system resource manager (SRM) calculates the paging rate using the alternate system’s residency time, rather than the execution time if PPGRT is specified.

For the XRF alternate system that has a low activity while in the surveillance phase, PPGRTR is a better choice because the target working set size is adjusted on the basis of page-faults per second, rather than page-faults per execution second.

During catchup and while tracking, the real storage needs of the XRF alternate CICS system are increased as it changes terminal session states and the contents of the TCT. At takeover, the real storage needs also increase as the alternate CICS system begins to switch terminal sessions and implement emergency restart. In order to ensure good performance and minimize takeover time, the target working set size should be increased. This can be done in several different ways, two of which are:

1. Parameter “b” in PWSS=(a,b) can be set to “*” which allows the working set size to increase without limit, if the maximum paging rate (parameter “d” in PPGRTR=(c,d)) is exceeded.
2. A command can be put in the CLT to change the alternate CICS system’s performance group at takeover to one which has different real storage isolation parameters specified.

If you set PWSS=(*,*), and PPGRTR=(1,2), this allows CICS to use as much storage as it wants when the paging rate is > 2 per second. The values depend very much on the installation and the MVS setup. The values suggested here assume that CICS is an important address space and therefore needs service to be resumed quickly.

For the definition and format of the storage isolation parameters in IEAIPSxx, see the *MVS/ESA Initialization and Tuning Guide*.

How implemented

See the *MVS/ESA Initialization and Tuning Guide*.

How monitored

Use RMF, the RMFMON command on TSO for additional information. The DISPLAY ACTIVE (DA) command on SDSF will give you an indication of the number of real pages used and the paging rate.

Increasing the CICS region size

If all other factors in a CICS system are kept constant, increasing the region size available to CICS allows an increase in the dynamic storage areas.

Changes to MVS and other subsystems over time generally reduce the amount of storage required below the 16MB line. Thus the CICS region size may be able to be increased when a new release of MVS or non-CICS subsystem is installed.

To get any further increase, operating-system functions and storage areas (such as the local shared queue area, LSQA), or other programs must be reduced. The LSQA is used by VTAM and other programs, and any increase in the CICS region size decreases the area available for the LSQA, SWA, and subpools 229 and 230. A shortage in these subpools can cause S80A, S40D, and S822 abends.

If you specify a larger region, the value of the relevant dsasize system initialization parameter must be increased or the extra space is not used.

How implemented

The region size is defined in the startup job stream for CICS. Other definitions are made to the operating system or through operating-system console commands.

To determine the maximum region size, determine the size of your private area from RMF II or one of the storage monitors available.

To determine the maximum region size you should allocate, use the following formula:

> Max region possible = private area size – system region size – (LSQA + SWA + subpools 229 and 230)

The remaining storage is available for the CICS region; for safety, use 80% or 90% of this number. If the system is static or does not change much, use 90% of this number for the REGION= parameter; if the system is dynamic, or changes frequently, 80% would be more desirable.

Note: You must maintain a minimum of 200KB of free storage between the top of the region and the bottom of the ESA high private area (the LSQA, the SWA, and subpools 229 and 230).

How monitored

Use RMF, the RMFMON command on TSO for additional information. For more information about RMF see “Resource measurement facility (RMF)” on page 32 or the *MVS/ESA RMF User's Guide*.

Giving CICS a high dispatching priority or performance group

Giving CICS a high dispatching priority causes the processor to be accessible more often when it is needed.

Performance groups in MVS are another way of giving CICS increased access to the processor. Putting CICS at a high dispatching priority or in a favorable performance group is most effective when CICS is processor-constrained.

The relative order of priority can be:

- VTAM
- Performance monitor
- Database
- CICS.

How implemented

Set the CICS priority above the automatic priority group (APG). See the *MVS/ESA Initialization and Tuning Guide* for further information.

There are various ways to assign CICS a dispatching priority. The best is through the ICS (PARMLIB member IEAICSxx). The ICS assigns performance group numbers and enforces assignments. The dispatching priorities are specified in PARMLIB member IEAIPSxx. Use APGRNG to capture the top ten priority sets (6 through 15). Specify a suitably high priority for CICS. There are priority levels that change dynamically, but we recommend a simple fixed priority for CICS. Use storage isolation only when necessary.

You cannot specify a response time, and you must give CICS enough resources to achieve good performance.

See the *MVS/ESA Initialization and Tuning Guide* for more information.

How monitored

Use either the DISPLAY ACTIVE (DA) command on SDSF or use RMF, the RMFMON command on TSO. For more information about RMF see “Resource measurement facility (RMF)” on page 32 or the *MVS/ESA RMF User's Guide*.

Using job initiators

The management of the MVS high private area can sometimes result in fragmentation and stranded subpools caused by large imbedded free areas known as “holes”.

Some fragmentation can also occur in a region when a job initiator starts multiple jobs without being stopped and then started again. If you define the region as having the maximum allowable storage size, it is possible to start and stop the job

the first time the initiator is used, but to have an S822 abend (insufficient virtual storage) the second time the job is started. This is because of the fragmentation that occurs.

In this situation, either the region has to be decreased, or the job initiator has to be stopped and restarted.

Two methods of starting the CICS job are available, to maximize the virtual storage available to the region. One is to start and stop the initiator with each initialization of CICS, executing CICS in a newly started initiator; and the other is to use the MVS START command.

If CICS is executed as an MVS-started task (using the MVS START command) instead of submitting it as a batch job, this not only ensures that a clean address space is used (reducing the possibility of an S822 abend), but also saves a significant amount of LSQA storage.

Effects

Some installations have had S822 abends after doing I/O generations or after adding DD statements to large applications. An S822 abend occurs when you request a REGION=nnnnK size that is larger than the amount available in the address space.

The maximum region size that is available is difficult to define, and is usually determined by trial and error. One of the reasons is that the size depends on the system generation and on DD statements.

At least two techniques can be used to reduce storage fragmentation:

1. **Dynamic allocation**. You might consider writing a “front-end” program that dynamically allocates the cataloged data sets for the step and then transfers control (XCTL) to CICS. The effect of this is that only one eligible device list (EDL) is used at a time.
2. **UNITNAME**. You might consider creating a new UNITNAME (via EDT-GEN or IOGEN). This UNITNAME could be a subset of devices known to contain the cataloged data set. By using the “unit override” feature of JCL, it could cause the EDL to be limited to the devices specified in the UNITNAME.

Limitations

Available virtual storage is increased by starting new initiators to run CICS, or by using MVS START. Startup time may be minimally increased.

How implemented

CICS startup and use of initiators are defined in an installation’s startup procedures.

How monitored

Part of the job termination message IEF374I 'VIRT=nnnnnK' shows you the virtual storage below the 16MB line, and another part 'EXT=nnnnnnnK' shows the virtual storage above the 16MB line.

Region exit interval (ICV)

When CICS cannot dispatch a task, either because there are no tasks in the system at that time, or because all tasks are waiting for data set or terminal I/O to finish, CICS issues an operating-system WAIT. The ICV, system initialization parameter (see also “Terminal scan delay (ICVTSD)” on page 204) controls the length of this wait (but bear in mind that any interrupt, for example, data set I/O or terminal I/O, before any of these expires, causes CICS to be dispatched). The keyword for ICV is time.

The ICV system initialization parameter specifies the maximum time in milliseconds that CICS releases control to the operating system when there are no transactions ready to resume processing. CICS issues a region wait in this case for the time specified in the ICV, system initialization parameter. If activity in the system causes CICS to be dispatched sooner, this parameter has no effect.

In general, ICV can be used in low-volume systems to keep part of the CICS management code paged in. Expiration of this interval results in a full terminal control table (TCT) scan in non-VTAM environments, and controls the dispatching of terminal control in VTAM systems with low activity. Redispatch of CICS by MVS/ESA after the wait may be delayed because of activity in the supervisor or in higher-priority regions, for example, VTAM. The ICV delay can affect the shutdown time if no other activity is taking place.

The value of ICV acts as a backstop for MROBTCH (see “VSAM resource usage (LSRPOOL)” on page 222).

Main effect

The region exit interval determines the maximum period between terminal control full scans. However, the interval between full scans in very active systems may be less than this, being controlled by the normally shorter terminal scan delay interval (see “Terminal scan delay (ICVTSD)” on page 204). In such systems, ICV becomes largely irrelevant unless ICVTSD has been set to zero.

Secondary effects

Whenever control returns to the task dispatcher from terminal control after a full scan, ICV is added to the current time of day to give the provisional due time for the next full scan. In idle systems, CICS then goes into an operating-system wait state, setting the timer to expire at this time. If there are application tasks to dispatch, however, CICS passes control to these and, if the due time arrives before CICS has issued an operating-system WAIT, the scan is done as soon as the task dispatcher next regains control.

In active systems, after the due time has been calculated by adding ICV, the scan may be performed at an earlier time by application activity (see “Terminal scan delay (ICVTSD)” on page 204).

Operating-system waits are not always for the duration of one ICV. They last only until some event ends. One possible event is the expiry of a time interval, but often CICS regains control because of the completion of an I/O operation. Before issuing the operating-system WAIT macro, CICS sets an operating-system timer, specifying the interval as the time remaining until the next time-dependent activity becomes

due for processing. This is usually the next terminal control scan, controlled by either ICV or ICVTSD, but it can be the earliest ICE expiry time, or even less.

In high-activity systems, where CICS is contending for processor time with very active higher-priority subsystems (VTAM, TSO, other CICS systems, or DB/DC), control may be seized from CICS so often that CICS always has work to do and never issues an operating-system WAIT.

Where useful

The region exit interval is useful in environments where batch or other CICS systems are running concurrently.

Limitations

Too low a value can impair concurrent batch performance by causing frequent and unnecessary dispatches of CICS by MVS/ESA. Too high a value can lead to an appreciable delay before the system handles time-dependent events (such as abends for terminal read or deadlock timeouts) after the due time.

A low ICV value does not prevent all CICS modules from being paged out. When the ICV time interval expires, the operating system dispatches CICS task control which, in turn, dispatches terminal control. CICS references only task control, terminal control, TCT, and the CSA. No other modules in CICS are referenced. If there is storage constraint they do not stay in real storage.

After the operating-system WAIT, redispatch of CICS may be delayed because of activity in the supervisor or in higher-priority regions such as VTAM, and so on.

The ICV delay can affect the shutdown time if no other activity is taking place.

Recommendations

The time interval can be any decimal value in the range from 100 through 3600000 milliseconds.

In normal systems, set ICV to 1000–10000 milliseconds, or more.

A low interval value can enable much of the CICS nucleus to be retained, and not be paged out at times of low terminal activity. This reduces the amount of paging necessary for CICS to process terminal transactions (thus representing a potential reduction in response time), sometimes at the expense of concurrent batch region throughput. Large networks with high terminal activity tend to drive CICS without a need for this value, except to handle the occasional, but unpredictable, period of inactivity. These networks can usually function with a large interval (10000 to 30000 milliseconds). After a task has been initiated, the system recognizes its requests for terminal services and the completion of the services, and overrides this maximum delay interval.

Small systems or those with low terminal activity are subject to paging introduced by other jobs running in competition with CICS. If you specify a low interval value, key portions of the CICS nucleus are referenced more frequently, thus reducing the probability of these pages being paged-out. However, the execution of the logic, such as terminal polling activity, without performing productive work might be considered wasteful.

You must weigh the need to increase the probability of residency by frequent but unproductive referencing, against the extra overhead and longer response times incurred by allowing the paging to occur. If you increase the interval size, more productive work is performed at the expense of performance if paging occurs during the periods of CICS activity.

Note: If the terminal control negative poll delay feature is used, the ICV value selected must not exceed the negative poll delay value. If the negative poll delay used is not zero, any ICV value may be used (see "Negative poll delay (NPDELAY)" on page 207).

How implemented

ICV is specified in the SIT or at startup, and can be changed using either the CEMT or EXEC CICS SET SYSTEM (time) command. It is defined in units of milliseconds, rounded down to the nearest multiple of ten. The default is 1000 (that is, one second; usually too low).

How monitored

The region exit interval can be monitored by the frequency of CICS operating-system WAITs that are counted in "Dispatcher statistics" on page 339.

Use of LLA (MVS/ESA library lookaside)

Modules loaded by CICS from the DFHRPL libraries may be managed by the MVS LLA (library lookaside) facility. LLA is designed to minimize disk I/O by keeping load modules in a VLF (virtual lookaside facility) dataspace and keeping a version of the library directory in its own address space.

LLA manages modules (system or application) whose library names you have put in the appropriate CSVLLA member in SYS1.PARMLIB.

There are two optional parameters in this member that affect the management of specified libraries:

FREEZE Tells LLA always to use its copy of the directory entries. This eliminates the I/O for directory reads.

NOFREEZE Tells LLA always to check the directory entry on DASD to see if the DASD version has been updated. If this is the case the version of the program on DASD is loaded.

| However, FREEZE and NOFREEZE are only relevant when LLACOPY is not used. When CICS issues a LOAD and specifies the directory entry (DE), it bypasses the LLA directory processing, but determines from LLA whether the program is already in VLF or must be fetched from DASD.

The use of LLA to manage a very busy DFHRPL library can show two distinct benefits:

1. Improved transaction response time
2. Better DASD utilization.

It is possible, as throughput increases, that DASD utilization actually decreases. This is due to LLA's observation of the load activity and its decisions about which modules to stage (keep) in the VLF dataspace.

LLA does not automatically stage all members that are fetched. LLA attempts to select those modules whose staging gives the best reductions in response time, contentions, storage cost, and an optional user-defined quantity.

In addition to any USER-defined CICS DFHRPL libraries, LLA also manages the system LNKLST. It is likely that staging some modules from the LNKLST could have more effect than staging modules from the CICS libraries. LLA makes decisions on what is staged to VLF only after observing the fetch activity in the system for a certain period. For this reason it is possible to see I/O against a program library even when it is managed by LLA.

Another contributing factor for continued I/O is the system becoming "MAXVIRT constrained", that is, the sum of bytes from the working set of modules is greater than the MAXVIRT parameter for the LLA class of VLF objects. You can increase this value by changing it in the COFVLF member in SYS1.PARMLIB. A value too small can cause excessive movement of that VLF object class; a value too large can cause excessive paging; both may increase the DASD activity significantly.

See *MVS/ESA Initialization and Tuning Guide* for information on LLA and VLF parameters.

Effects of LLACOPY

CICS can use one of two methods for locating modules in the DFHRPL concatenation. Either a BLDL macro or a LLACOPY macro is issued to return the directory information to pass to the load request. Which macro is issued is dependant upon the LLACOPY system initialization parameter and the reason for the locate of the module.

The LLACOPY macro is used to update the LLA-managed directory entry for a module or a list of modules. If a module which is LLA managed has an LLACOPY issued against it, it results in a BLDL with physical I/O against the DCB specified. If the directory information does not match that which is stored within LLA, the LLA tables are then updated, keeping both subsystems synchronized. While this activity takes place an ENQ for the resource SYSZLLA1.update is held. This is then unavailable to any other LLACOPY request on the same MVS system and therefore another LLACOPY request is delayed until the ENQ is released.

The BLDL macro also returns the directory information. When a BLDL is issued against an LLA managed module, the information returned will be from the LLA copy of the directory, if one exists. It will not necessarily result in physical I/O to the dataset and may therefore be out of step with the actual dataset. BLDL does not require the SYSZLLA1.update ENQ and is therefore less prone to being delayed by BLDLs on the same MVS system.

The SIT Parameter LLACOPY

If you code LLACOPY=YES, the default, CICS issues a LLACOPY macro each time a module is located from the RPL dataset. This is done either on the first ACQUIRE or on any subsequent NEWCOPY or PHASEIN requests. This ensures that CICS always obtains the latest copy of any LLA-managed modules. There is a small chance of delay because of a failure to obtain an ENQ while another LLACOPY completes and there is some extra pathlength involved in maintaining the LLA tables.

If you code LLACOPY=NO, CICS never issues an LLACOPY macro. Instead, each time the RPL dataset is searched for a module a BLDL is issued.

If a NEWCOPY of an LLA-managed module is required, a MODIFY LLA,REFRESH or F LLA,REFRESH must be issued before the NEWCOPY is performed within CICS.

If you code LLACOPY=NEWCOPY, CICS issues the LLACOPY macro when loading a module as a result of a NEWCOPY or PHASEIN request. A BLDL macro is issued on the first use of the module unless it is due to one of the previously mentioned requests. This could mean an out of date version of a module is loaded upon its first use, but the latest version would be used after a NEWCOPY or PHASEIN.

DASD tuning

The main solutions to DASD problems are to:

- Reduce the number of I/O operations
- Tune the remaining I/O operations
- Balance the I/O operations load.

Reducing the number of I/O operations

The principal ways of reducing the number of I/O operations are to:

- Allocate VSAM Hiperspace buffers
- Allocate additional address space buffers
- Use data tables when appropriate
- Use or increase the use of main temporary storage
- Eliminate or minimize program compression
- Review and improve the design of applications run on CICS
- Make use of a DASD controller cache, but only if data set placement tuning has been done
- Minimize CI/CA splits by:
  - Allocating ample free space (free space can be altered by key range during load)
  - Timely reorganizations of disk storage.

Tuning the I/O operations

This can reduce service time. The principal ways of tuning the I/O operations are to:

- Specify the correct CI size. This has an effect on:
  - The space used on the volume
  - Transfer time
  - Storage requirements for buffers
  - The type of processing (direct or sequential).
- Use imbeds or replication for VSAM data sets
- Specify the location of the VTOC correctly.

- Take care over data set placement within the volume.
- Use an appropriately-fast device type and, if necessary, use a cache memory (but only if data set placement tuning has been done and if there are sufficient channels to handle the device speed).

Balancing I/O operations

This can reduce queue time. The principal ways of balancing I/O operations are to:

- Spread a high-use data set across multiple volumes.
- Minimize the use of shared DASD volumes between multiple processors.
- Place batch files and online files on separate volumes, especially:
  - Spool files
  - Sort files
  - Assembler or compiler work files
  - Page data sets.
- Place index and data on separate volumes (for VSAM KSDS files).
- Place concurrently-used files on separate volumes. For example, a CICS journal should be the only data set in use on its volume.

Take the following figures as guidelines for best DASD response times for online systems:

- Channel busy: less than 30% (with CHP ids this can be higher)
- Device busy: less than 35% for randomly accessed files
- Average response time: less than 20 milliseconds.

Aim for multiple paths to disk controllers because this allows dynamic path selection to work.

Chapter 17. Networking and VTAM

This chapter includes the following topics:

Terminal input/output area (TYPETERM IOAREALEN or TCT TIOAL)

If you are using VTAM, the CEDA DEFINE TYPETERM IOAREALEN command determines the initial size of the terminal input/output area (TIOA) to be passed onto a transaction for each terminal. The syntax for IOAREALEN is ({0|value1},{0|value2}). This operand is used only for the first input message for all transactions.

For TCAM, the DFHTCT TYPE=TERMINAL TIOAL=value macro, is the only way to adjust this value.

One value defining the minimum size is used for non-SNA devices, while two values specifying both the minimum and maximum size are used for SNA devices.

This book does not discuss the performance aspects of the CICS/ESA Front End Programming Interface. See the *CICS/ESA Front End Programming Interface User's Guide* for more information.

Effects

When value1,0 is specified for IOAREALEN, value1 is the minimum size of the terminal input/output area that is passed to an application program when a RECEIVE command is issued. If the size of the input message exceeds value1, the area passed to the application program is the size of the input message.

When value1, value2 is specified, value1 is the minimum size of the terminal input/output area that is passed to an application program when a RECEIVE command is issued. Whenever the size of the input message exceeds value1, CICS will use value2. If the input message size exceeds value2, the node abnormal condition program sends an exception response to the terminal.

If you specify ATI(YES), you must specify an IOAREALEN of at least one byte.

For TCAM supported devices, if the TIOAL operand is omitted, it defaults to the INAREAL length value in the TCT TYPE=LINE operand. Do not omit the TIOAL operand for remote terminals.

Limitations

Real storage can be wasted if the IOAREALEN (value1) or TIOAL value is too large for most terminal inputs in the network. If IOAREALEN (value1) or TIOAL is smaller than most initial terminal inputs, excessive GETMAIN requests can occur, resulting in additional processor requirements, unless IOAREALEN(value1) or TIOAL is zero.

Recommendations

IOAREALEN(value1) or TIOAL should be set to a value that is slightly larger than the average input message length for the terminal. The maximum value that may be specified for IOAREALEN/TIOAL is 32767 bytes.

If a value of nonzero is required, the best size to specify is the most commonly encountered input message size. A multiple of 64 bytes minus 21 allows for SAA requirements and ensures good use of operating system pages.

For VTAM, you can specify two values if inbound chaining is used. The first value should be the length of the normal chain size for the terminal, and the second value should be the maximum size of the chain. The length of the TIOA presented to the task depends on the message length and the size specified for the TIOA. (See the example in Figure 32.)

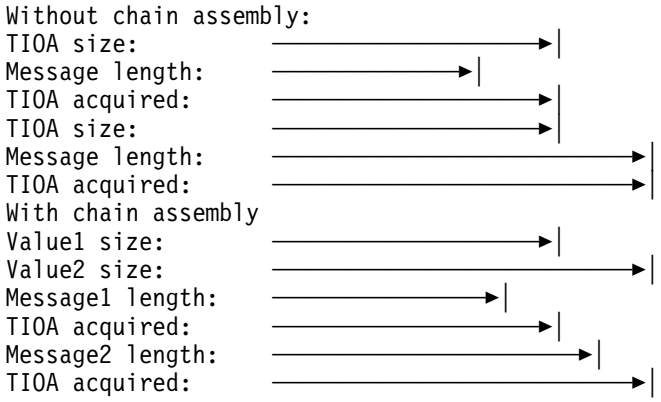


Figure 32. Message length and terminal input/output area length

Avoid specifying too large a value1, for example, by matching it to the size of the terminal display screen. This area is used only as input. If READ with SET is specified, the same pointer is used by applications for an output area.

If too small a value is specified for value1, extra processing time is required for chain assembly, or data is lost if inbound chaining is not used.

In general, a value of zero is best because it causes the optimum use of storage and eliminates the second GETMAIN request. If automatic transaction initiation (ATI) is used for that terminal, a minimum size of one byte is required.

The second value for SNA devices is used to prevent terminal streaming, and so should be slightly larger than the largest possible terminal input in the network. If a message larger than this second value is encountered, a negative response is returned to the terminal, and the terminal message is discarded.

How implemented

For VTAM, the TIOA value is specified in the CEDA DEFINE TYPETERM IOAREALEN keyword.

For TCAM, the TIOAL value can be specified in the terminal control table (TCT) TYPE=TERMINAL operand. TIOAL defaults to the INAREAL value specified in the TCT TYPE=LINE operand.

How monitored

RMF and NetView Performance Monitor (NPM) can be used to show storage usage and message size characteristics in the network.

Receive-any input areas (RAMAX)

The system initialization parameter, RAMAX, specifies the size in bytes of the I/O area that is to be allocated for each VTAM receive-any operation. These storage areas are called receive-any input areas (RAIAs), and are used to receive the first terminal input for a transaction from VTAM. All input from VTAM comes in request/response units (RUs).

Storage for the RAIAs, which is above the 16MB line, is allocated by the CICS terminal control program during CICS initialization, and remains allocated for the entire execution of the CICS job step. The size of this storage is the product of the RAPOOL and RAMAX system initialization parameters.

Effects

VTAM attempts to put any incoming RU into the initial receive-any input area, which has the size of RAMAX. If this is not large enough, VTAM indicates that and also states how many extra bytes are waiting that cannot be accommodated.

RAMAX is the largest size of any RU that CICS can take directly in the receive-any command, and is a limit against which CICS compares VTAM's indication of the overall size of the RU. If there is more, VTAM saves it, and CICS gets the rest in a second request.

With a small RAMAX, you reduce the virtual storage taken up in RAIAs but risk more processor usage in VTAM retries to get any data that could not fit into the RAIA.

For many purposes, the default RAMAX value of 256 bytes is adequate. If you know that many incoming RUs are larger than this, you can always increase RAMAX to suit your system.

For individual terminals, there are separate parameters that determine how large an RU is going to be from that device. It makes sense for RAMAX to be at least as large as the largest CEDA SENDSIZE for any frequently-used terminals.

Where useful

You can use the RAMAX system initialization parameter in any networks that use the VTAM access method for terminals.

Limitations

Real storage can be wasted with a high RAMAX value, and additional processor time can be required with a low RAMAX value. If the RAMAX value is set too low, extra processor time is needed to acquire additional buffers to receive the remaining data. Because most inputs are 256 bytes, this should normally be specified.

Recommendations

Code RAMAX with the size in bytes of the I/O area allocated for each receive-any request issued by CICS. The maximum value is 32767.

Set RAMAX to be slightly larger than your CICS system input messages. If you know the message length distribution for your system, set the value to accommodate the majority of your input messages.

In any case, the size required for RAMAX need only take into account the first (or only) RU of a message. Thus, messages sent using SNA chaining do not require RAMAX based on their overall chain length, but only on the size of the constituent RUs.

| Receive-any input areas are taken from a fixed length subpool of storage. A size
| of 2048 may appear to be adequate for two such areas to fit on one 4KB page, but
| only 4048 bytes are available in each page, so only one area fits on one page. A
| size of 2024 should be defined to ensure that two areas, including page headers, fit
| on one page.

How implemented

RAMAX is a system initialization parameter.

How monitored

The size of RUs or chains in a network can be identified with a VTAM line or buffer trace. The maximum size RUs are defined in the CEDA SENDSIZE keyword.

Receive-any pool (RAPOOL)

The RAPOOL system initialization parameter specifies the number of concurrent receive-any requests that CICS is to process from VTAM. RAPOOL determines how many receive-any buffers there are at any time and, therefore, if VTAM has a lot of input simultaneously, it enables VTAM to put all the messages directly into CICS buffers rather than possibly having to store them itself elsewhere. The first operand (value1) is for non-HPO systems, the second operand (value2) is for HPO systems.

The HPO value for the non-HPO operand is derived according to the formula
shown in the *CICS/ESA System Definition Guide*. The second operand (value2) for
HPO systems is used with minimal adjustment by the formula.

Effects

Initially, task input from a terminal or session is received by the VTAM access method and is passed to CICS if CICS has a receive-any request outstanding.

For each receive-any request, a VTAM request parameter list (RPL), a receive-any control element (RACE), and a receive-any input area (RAIA)—the value specified by RAMAX (see “Receive-any input areas (RAMAX)”) are set aside. The total area set aside for VTAM receive-any operations is:

(maximum RAIA size + RACE size + RPL size) * RAPOOL

If HPO=YES, then both RACE and RPL are above the 16MB line.

See page 195 for RAIA considerations.

In general, input messages up to the value specified in RAPOOL are all processed in one dispatch of the terminal control task. Because the processing of a receive-any request is a short operation, at times more messages than the RAPOOL value may be processed in one dispatch of terminal control. This happens when a receive-any request completes before the terminal control program has finished processing and there are additional messages from VTAM.

VTAM receive-any processing is for the first terminal message in a transaction, so RAPOOL has no effect on further inputs for conversational tasks. Those additional inputs are processed with VTAM receive-specific requests.

The pool is used only for the first input to start a task; it is not used for output or conversational input. VTAM posts the event control block (ECB) associated with the receive any input area. CICS then moves the data to the terminal I/O area (TIOA) ready for task processing. The RAIA is then available for reuse.

Where useful

Use the RAPOOL operand in networks that use the VTAM access method for terminals.

Limitations

If the RAPOOL value is set too low, this can result in terminal messages not being processed in the earliest dispatch of the terminal control program, thereby inducing transaction delays during high-activity periods. For example, if you use the default and five terminal entries want to start up tasks, three tasks may be delayed for at least the time required to complete the VTAM receive-any request and copy the data and RPL. In general, no more than 5 to 10% of all receive-any processing should be at the RAPOOL ceiling, with none being at the RAPOOL ceiling if there is sufficient storage.

If the RAPOOL value is set too high, this can use excessive virtual storage, but does not affect real storage because the storage is not page-fixed and is therefore paged out.

Recommendations

Whether RAPOOL is significant or not depends on the environment of the CICS system: whether, for example, HPO is being used.

In some cases, it may sometimes be more economical for VTAM to store the occasional peak of messages in its own areas rather than for CICS itself to have a large number of RAIAs, many of which are unused most of the time.

Furthermore, there are situations where CICS reissues a receive-any as soon as it finds one satisfied. It thereby uses the same element over and over again in order to bring in any extra messages that are in VTAM.

CICS maintains a VTAM RECEIVE ANY for n of the RPLs, where n is either the RAPOOL value, or the MXT value minus the number of currently active tasks, whichever is the smaller. See the *CICS/ESA System Definition Guide* for more information about these SIT parameters.

A general recommendation is to code RAPOOL with the number of fixed request parameter lists (RPLs) that you require. When it is not at MXT, CICS maintains a receive-any request for each of these RPLs. The number of RPLs that you require depends on the expected activity of the system, the average transaction lifetime, and the MXT specified.

The RAPOOL value you set depends on the number of sessions, the number of terminals, and the ICVTSD value (see page 204) in the system initialization table (SIT). Initially, for non-HPO systems, you should set RAPOOL to 1.5 times your peak ***local*** [2] transaction rate per second plus the autoinstall rate. This can then be adjusted by analyzing the CICS VTAM statistics and by resetting the value to the maximum RPLs reached.

For HPO systems, a small value (<= 5) is usually sufficient if specified through the value2 in the RAPOOL system initialization parameter. Thus, RAPOOL=20, for example, is specified either RAPOOL=(20) or RAPOOL=(20,5) to achieve the same effect.

How implemented

RAPOOL is a system initialization parameter.

How monitored

The CICS VTAM statistics contain values for the maximum number of RPLs posted on any one dispatch of the terminal control program, and the number of times the RPL maximum was reached. This maximum value may be greater than the RAPOOL value if the terminal control program is able to reuse an RPL during one dispatch. See “VTAM statistics” on page 51 for more information.

[2] The RAPOOL figure does not include MRO sessions, so you should set RAPOOL to a low value in application- or file-owning regions (AORs or FORs).

High performance option (HPO) with VTAM

The MVS high performance option (HPO) can be used for processing VTAM requests. The purpose of HPO is to reduce the transaction pathlength through VTAM.

Effects

HPO bypasses some of the auditing functions performed by MVS on I/O operations, and implements service request block (SRB) scheduling. This shortens the instruction pathlength and allows some concurrent processing on MVS images for the VTAM operations because of the SRB scheduling. This makes it useful in a multi processor environment, but not in a single processor environment.

Limitations

HPO requires CICS to be authorized, and some risks with MVS integrity are involved because a user-written module could be made to replace one of the CICS system initialization routines and run in authorized mode. This risk can be reduced by RACF protecting the CICS SDFHAUTH data set.

Use of HPO saves processor time, and does not increase real or virtual storage requirements or I/O contention. The only expense of HPO is the potential security exposure due to running in authorized mode. However, note that certain other CICS functions also require authorization. These functions are those that use the CICS SVCs.

You should be careful of using HPO in a system that is being used for early testing of a new application or CICS code (a new release or PUT). Much of the pathlength reduction is achieved by bypassing control block verification code in VTAM. In a situation where untested code might possibly corrupt the control blocks that CICS passes to VTAM, there is a possibility that problem determination may be more difficult and VTAM might even be caused to abend.

Recommendations

The general recommendation is that all CICS/ESA VTAM users should use HPO. It is totally application-transparent and introduces no function restrictions while providing a reduced pathlength through VTAM. In the case of VTAM, the reduced auditing does not induce any integrity loss for the messages.

How implemented

The SVCs and use of HPO are specified in the system initialization table (SIT) and, if the default SVC numbers are acceptable, no tailoring of the system is required.

How monitored

There is no direct measurement of HPO. One way to tell if it is working is to take detailed measurements of processor usage with HPO turned on (SIT option) and with it turned off. Depending on the workload, you may not see much difference. Another way to tell if it is working is that you may see a small increase in the SRB scheduling time with HPO turned on.

RMF can give general information on processor usage. An SVC trace can show how HPO was used.

SNA transaction flows (MSGINTEG, PROTECT, and ONEWTE)

Within CICS, the MSGINTEG and PROTECT options can be used to control the communication requests and responses that are exchanged between the terminals in a network and the VTAM and NCP communications programs.

Effects

One of the options in Systems Network Architecture (SNA) is whether the messages exchanged between CICS and a terminal are to be in definite or exception response mode. Definite response mode requires both the terminal and CICS to provide acknowledgment of receipt of messages from each other on a one-to-one basis.

SNA also ensures message delivery through synchronous data link control (SDLC), so definite response is not normally required. Specifying either message integrity (MSGINTEG) or message protection (PROTECT) causes the sessions for which it is specified to operate in definite response mode.

In other cases, the session between CICS and a terminal operates in exception response mode, and this is the normal case. Specifying PROTECT adds the overhead of writing the messages to the CICS system log as well.

In SNA, transactions are defined within brackets. A begin bracket (BB) command defines the start of a transaction, and an end bracket (EB) command defines the end of that transaction. Unless CICS knows ahead of time that a message is the last of a transaction, it must send an EB separate from the last message if a transaction terminates. The EB is an SNA command, and can be sent with the message, eliminating one required transmission to the terminal.

Specifying the ONEWTE option for a transaction implies that only one output message is to be sent to the terminal by that transaction, and allows CICS to send the EB along with that message. Only one output message is allowed if ONEWTE is specified and, if a second message is sent, the transaction is abended.

The second way to allow CICS to send the EB with a terminal message is to code the LAST option on the last terminal control or basic mapping support SEND command in a program. Multiple SEND commands can be used, but the LAST option must be coded for the final SEND in a program.

The third (and most common) way is to issue SEND without WAIT as the final terminal communication. The message is then sent as part of task termination.

You have the following options:

- Specifying neither MSGINTEG nor PROTECT
- Specifying MSGINTEG only (which simply asks for definite response to be forced)
- Specifying PROTECT only (which not only forces MSGINTEG but also causes logging of sequence numbers before and after, and therefore causes more overhead in the processing of messages).

Where useful

The above options can be used in all CICS systems that use VTAM.

Limitations

The MSGINTEG option causes additional transmissions to the terminal. Transactions remain in CICS for a longer period, and tie up virtual storage and access to resources (primarily enqueues). MSGINTEG is required if the transaction must know that the message was delivered.

The PROTECT option incurs the overhead of MSGINTEG and adds the overhead of logging, but is required for message resynchronization and re-presentation on emergency restart in the case of transactions using CICS as an intelligent programmed SNA controller.

When MSGINTEG is specified, the TIOA remains in storage until the response is received from the terminal. If PROTECT is specified, the task is delayed until the response is received. Either option can increase the virtual storage requirements for the CICS/ESA region because of the longer duration of the storage needs.

How implemented

With resource definition online (RDO) using the CEDA transaction, protection can be specified in the profile definition by means of the MSGINTEG, ONEWTE, and PROTECT options. MSGINTEG and PROTECT options are used with SNA LUs only. See the *CICS/ESA Resource Definition Guide* for more information about the profile definition.

How monitored

You can monitor the use of the above options from a VTAM trace by examining the exchanges between terminals and CICS and, in particular, by examining the contents of the request/response header (RH).

SNA chaining (TYPETERM RECEIVESIZE, BUILDCHAIN, and SENDSIZE)

Systems Network Architecture (SNA) allows terminal messages to be chained, and lets large messages be split into smaller parts while still logically treating the multiple message as a single message.

Input chain size and characteristics are normally dictated by the hardware requirements of the terminal in question, and so the CEDA BUILDCHAIN and RECEIVESIZE keywords have no default values. The size of an output chain is specified by the CEDA SENDSIZE keyword.

Effects

Because the network control program (NCP) also segments messages into 256-byte blocks for normal LU Type 0, 1, 2, and 3 devices, a SENDSIZE value of zero eliminates the overhead of output chaining. A value of 1536 is required for local devices of this type.

If you specify the CEDA SENDSIZE keyword for intersystem communication (ISC) sessions, this must match the CEDA RECEIVESIZE keyword in the other system.

The CEDA SENDSIZE keyword or TCT BUFFER operand controls the size of the SNA element that is to be sent, and the CEDA RECEIVESIZEs need to match so that there is a corresponding buffer of the same size able to receive the element.

If you specify BUILDCHAIN(YES), CICS assembles a complete chain of elements before passing them to an application. If you do not specify BUILDCHAIN(YES), each individual RU is passed to an individual receive-any in the application. With SNA/3270 BMS does not work correctly if you do not specify BUILDCHAIN(YES).

If you are dealing with very large inbound elements that exceed a maximum of 32KB, you cannot use the BUILDCHAIN keyword or CHNASSY operand. You must use multiple individual RUs, and this extends the transaction life in the system.

Where useful

Chaining can be used in systems that use VTAM and SNA terminals of types that tolerate chaining.

Limitations

If you specify a low CEDA SENDSIZE value, this causes additional processing and real and virtual storage to be used to break the single logical message into multiple parts.

Chaining may be required for some terminal devices. Output chaining can cause flickering on display screens, which can annoy users. Chaining also causes additional I/O overhead between VTAM and the NCP by requiring additional VTAM subtasks and STARTIO operations. This additional overhead is eliminated with applicable ACF/VTAM releases by making use of the large message performance enhancement option (LMPEO).

Recommendations

The CEDA RECEIVESIZE value for IBM 3274-connected display terminals should be 1024; for IBM 3276-connected display terminals it should be 2048. These values give the best line characteristics while keeping processor usage to a minimum.

How implemented

Chaining characteristics are specified in the CEDA DEFINE TYPETERM statement with the SENDSIZE, BUILDCHAIN, and RECEIVESIZE keywords.

How monitored

Use of chaining and chain size can be determined by examining a VTAM trace. You can also use the CICS internal and auxiliary trace facilities, in which the VIO ZCP trace shows the chain elements. Some of the network monitor tools such as NetView Performance Monitor (NPM) give this data.

Number of concurrent logon/logoff requests (OPNDLIM)

The OPNDLIM operand defines the number of concurrent VTAM logons and logoffs that are to be processed by CICS. In systems running ACF/VTAM Release 3.2 and later, with the prerequisite PTF (No UN49759) applied for APAR PN36044, this operand is not necessary and will be ignored. In all other instances this system initialization parameter limits the number of concurrent logon OPNDST and logoff CLSDST requests. The smaller this value, the smaller the amount of storage that is required during the open and close process.

Each concurrent logon/logoff requires storage in the CICS dynamic storage areas for the duration of that processing.

Effects

Particularly when logons are being done automatically with either the CICS CONNECT=AUTO facility or the VTAM LOGAPPL facility, large numbers of logons can occur at CICS startup or restart times. In systems running ACF/VTAM with a release prior to 3.2 this can require significant amounts of storage, which can be reduced with the OPNDLIM operand. In ACF/VTAM Release 3.2 and later systems with the prerequisite PTF (No UN49759) applied for APAR PN36044, this operand is not necessary and will be ignored.

If an automatic logon facility is required, the LOGAPPL facility offers two advantages. It requires approximately 3500 bytes less storage in VTAM than the CONNECT=AUTO facility, and it logs terminals back on to CICS each time the device is activated to VTAM, rather than only at CICS initialization.

Where useful

The OPNDLIM system initialization parameter can be used in CICS systems that use VTAM as the terminal access method.

The OPNDLIM system initialization parameter can also be useful if there are times when all the user community tends to log on or log off at the same time, for example, during lunch breaks.

Limitations

If too low a value is specified for OPNDLIM, real and virtual storage requirements are reduced within CICS and VTAM buffer requirements may be cut back, but session initializations and terminations take longer.

Recommendations

Use the default value initially and make adjustments if statistics indicate that too much storage is required in your environment or that the startup time (DEFINE TYPETERM AUTOCONNECT keyword in CEDA) is excessive.

OPNDLIM should be set to a value not less than the number of LUs connected to any single VTAM line.

How implemented

OPNDLIM is a system initialization parameter.

How monitored

Logon and logoff activities are not reported directly by CICS or any measurement tools, but can be analyzed using the information given in a VTAM trace or VTAM display command.

Terminal scan delay (ICVTSD)

The terminal scan delay (ICVTSD) system initialization parameter determines the frequency with which CICS attempts to process terminal output requests.

In general, this value defines the time that the terminal control program must wait to process:

- Non-VTAM terminal I/O requests with WAIT specified
- Non-VTAM output deferred until task termination
- Automatic transaction initiation (ATI) requests
- VTAM terminal management, including output request handling, in busy CICS systems with significant application task activity.

This last case arises from the way that CICS scans active tasks.

On CICS non-VTAM systems, the delay value specifies how long the terminal control program must wait after an application terminal request, before it carries out a TCT scan. The value thus controls batching and delay in the associated processing of terminal control requests. In a low-activity system, it controls the dispatching of the terminal control program.

The batching of requests reduces processor time at the expense of longer response times. On CICS VTAM systems, it influences how quickly the terminal control program completes VTAM request processing, especially when the MVS high performance option (HPO) is being used.

Effects

VTAM

In VTAM networks, a low ICVTSD value does not cause full TCT scans because the output from VTAM terminals is processed from the activate queue chain, and only those terminal entries are scanned.

With VTAM terminals, CICS uses bracket protocol to indicate that the terminal is currently connected to a transaction. The bracket is started when the transaction is initiated, and ended when the transaction is terminated. This means that there could be two outputs to the terminal per transaction: one for the data sent and one when the transaction terminates containing the end bracket. In fact, only one output is sent (except for WRITE/SEND with WAIT and definite response). CICS holds the output data until the next terminal control request or termination. In this way it saves processor cycles and line utilization by sending the message and end bracket or change direction (if the next request was a READ/RECEIVE) together in

the same output message (PIU). When the system gets very busy, terminal control is dispatched less frequently and becomes more dependent upon the value specified in ICVTSD. Because CICS may not send the end bracket to VTAM for an extended period of time, the life of a transaction can be extended. This keeps storage allocated for that task for longer periods and potentially increases the amount of virtual storage required for the total CICS dynamic storage areas.

Setting ICVTSD to zero can overcome this effect.

Non-VTAM

ICVTSD is the major control on the frequency of full terminal control table (TCT) scanning of non-VTAM terminals. In active systems, a full scan is done approximately once every ICVTSD. The average extra delay before sending an output message should be about half this period.

In non-VTAM networks, partial scans occur for other reasons, such as an input arriving from a terminal, and any outputs for that line are processed at the same time. For that reason, a value of between 0.5 and one second is normally a reasonable setting for non-VTAM networks.

CICS scans application tasks first, unless there is an ICVTSD-driven scan. In a highly utilized system, input and output messages may be unreasonably delayed if too large a ICVTSD value is specified.

All networks

In reasonably active systems, a nonzero ICVTSD virtually replaces ICV (see page 187) because the time to the next TCT full scan (non-VTAM) or sending of output requests (VTAM) is the principal influence on operating system wait duration.

The ICVTSD parameter can be changed in the system initialization table (SIT) or through JCL parameter overrides. If you are having virtual storage constraint problems, it is highly recommended that you reduce the value specified in ICVTSD. A value of zero causes the terminal control task to be dispatched most frequently. If you also have a large number of non-VTAM terminals, this may increase the amount of nonproductive processor cycles. A value of 100—300 milliseconds may be more appropriate for that situation. In a pure VTAM environment, however, the overhead is not significant, unless the average transaction has a very short pathlength, and ICVTSD should be set to zero for a better response time and best virtual storage usage.

Where useful

The ICVTSD system initialization parameter can be used in all except very low-activity CICS systems.

Limitations

In TCAM systems, a low ICVTSD value can cause excessive processor time to be used in slower processor units, and can delay the dispatch of user tasks because too many full TCT scans have to be done. A high ICVTSD value can increase response time by an average of one half of the ICVTSD value, and can tie up resources owned by the task because the task takes longer to terminate. This applies to conversational tasks.

In VTAM systems, a low value adds the overhead of scanning the activate queue TCTTE chain, which is normally a minor consideration. A high value in high-volume systems can increase task life and tie up resources owned by that task for a longer period of time; this can be a significant consideration.

A low, nonzero value of ICVTSD can cause CICS to be dispatched more frequently, which increases the overhead of performance monitoring.

Recommendations

Set ICVTSD to a value less than the region exit time interval (ICV), which is also in the system initialization table (see page 185). Use the value of zero in an environment that contains only VTAM terminals and consoles, unless your workload consists of many short transactions. ICVTSD=0 in a VTAM terminal-only environment is not recommended for a CICS workload consisting of low terminal activity but with high TASK activity. Periods of low terminal activity can lead to delays in CSTP being dispatched. Setting ICVTSD=100-500 resolves this by causing CSTP to be dispatched regularly. For non-VTAM systems specify the value of zero only for small networks (1 through 30 terminals).

For almost all systems that are not “pure” VTAM, the range should be somewhere in the region of 100 milliseconds to 1000 milliseconds. ICVTSD can be varied between, say, 300 and 1000 milliseconds without a very significant effect on the response time, but increasing the value decreases the processor overhead. An ICVTSD larger than 1000 milliseconds may not give any further improvement in processor usage, at a cost of longer response times.

If ICVTSD is reduced, and, if there is ample processor resource, a small reduction in response time can be achieved. If you go below 250 milliseconds, any improvement in response time is likely to seem negligible to the end user and would have an increased effect on processor usage.

The recommended absolute minimum level, for systems that are not “pure” VTAM, is approximately 250 milliseconds or, in really high-performance, high-power systems that are “pure” VTAM, 100 milliseconds.

How implemented

The ICVTSD system initialization parameter is defined in units of milliseconds. Use the commands CEMT or EXEC CICS SET SYSTEM SCANDELAY (nnnn) to reset the value of ICVTSD.

How monitored

Use RMF to monitor task duration and processor requirements. The dispatcher domain statistics reports the value of ICVTSD.

Negative poll delay (NPDELAY)

NPDELAY in the TCT TYPE=LINE macro helps reduce unsolicited-input errors on TCAM lines.

NPDELAY and unsolicited-input messages in TCAM

Any CICS users who do not want unsolicited-input messages to be discarded should consider using the optional NPDELAY operand for the DFHTCT TYPE=LINE macro used to define each of the TCAM queues. This allows you to define a time interval during which CICS suspends the reading of messages from the respective TCAM queue following the receipt of unsolicited input. Upon completion of the preceding transaction associated with the same terminal as the unsolicited message, within the NPDELAY-defined interval, processing resumes normally.

If the preceding transaction fails to terminate during the NPDELAY interval, the X'87' unsolicited-input error condition is raised.

When NPDELAY is used, it is frequently advisable to define several input queues to CICS, each defined by a separate LINE entry but with each entry naming the same corresponding output queue using the OUTQ parameter. An equivalent number of queues must be defined to TCAM with the TPROCESS macro. This set of queues can then be processed as a “cascade” list within TCAM.

When several queues are defined for TCAM-to-CICS processing, CICS can suspend the acceptance of input messages from one or more of the queues without completely stopping the flow of input from TCAM to CICS.

Choosing an appropriate value for NPDELAY is a matter of tuning. Even with the “cascade” list approach, some messages may be held up behind an unsolicited message. The objective should be to find the minimum value that can be specified for NPDELAY which is sufficient to eliminate the unsolicited-input errors.

Compression of output terminal data streams

For output messages, CICS provides user exits with access to the entire output data stream. User code can be written to remove redundant characters from the data stream before the data stream is sent to the terminal. This technique can produce a dramatic improvement in response times if the proportion of characters not needed is large, because telecommunication links are usually the slowest paths in the network.

Limitations

Some additional processor cycles are required to process the exit code, and the coding of the exit logic also requires some effort. Use of a compression exit reduces the storage requirements of VTAM or TCAM and NCP, and reduces line transmission time.

Recommendations

The simplest operation is to replace redundant characters, especially blanks, with a repeat-to-address sequence in the data stream for 3270-type devices.

Note: The repeat-to-address sequence is not handled very quickly on some types of 3270 cluster controller. In some cases, alternatives may give superior performance. For example, instead of sending a repeat-to-address sequence for a series of blanks, you should consider sending an ERASE and then set-buffer-address sequences to skip over the blank areas. This is satisfactory if nulls are acceptable in the buffer as an alternative to blanks.

Another technique for reducing the amount of data transmitted is to turn off any modified data tags on protected fields in an output data stream. This eliminates the need for those characters to be transmitted back to the processor on the next input message, but you should review application dependencies on those fields before you try this.

There may be other opportunities for data compression in individual systems, but you may need to investigate the design of those systems thoroughly before you can implement them.

How implemented

The global user exits used to compress terminal messages are the XZCOUT1 exit for VTAM devices, and the XTCTOUT exit for TCAM-supported devices. See the *CICS/ESA Customization Guide* for programming information.

How monitored

The contents of output terminal data streams can be examined in either a VTAM or TCAM trace.

Automatic installation of terminals

During autoinstall processing, CICS obtains storage from the control subpool in the extended CICS dynamic storage area (ECDSA), to handle each autoinstall request. The amount of virtual storage obtained is mainly determined by the length of the CINIT request unit, which varies for different LU types. For a typical autoinstall request from an LU6.2 terminal, the amount of dynamic virtual storage obtained is between 120 to 250 bytes.

Overall, the principal consumer of CICS resource in autoinstall processing is the autoinstall task (CATA) itself. If, for some reason, the autoinstall process is not proceeding at the rate expected during normal operations, there is a risk that the system could be filled with CATA transaction storage.

Maximum concurrent autoinstalls (AIQMAX)

This system initialization parameter codes the maximum number of devices that can be queued concurrently for autoinstall.

The AIQMAX value does not limit the total number of devices that can be autoinstalled.

The restart delay parameter (AIRDELAY)

This system initialization parameter specifies whether you want autoinstalled terminal definitions to be retained by CICS across a restart. The value of the restart delay is specified as “hhmmss” and the default is “000700”, which is seven minutes. This means that if a terminal does not log on to CICS within seven minutes after an emergency restart, its terminal entry is scheduled for deletion.

Setting the restart delay to zero means that you do not want CICS to re-install the autoinstalled terminal entries from the global catalog after restart. In this case, CICS does not write the terminal entries to the catalog while the terminal is being autoinstalled. This can have positive performance effects on the following processes:

Autoinstall: By eliminating the I/O activity, autoinstall has a shorter pathlength and becomes more processor-intensive. So, in general, the time taken to autoinstall a terminal is reduced. However, the response time of other tasks may increase slightly because CATA has a high priority and does not have to wait for as much I/O activity.

Emergency and warm restart: When no autoinstalled terminal entries are cataloged, CICS has to restore fewer entries from the restart data set during warm or emergency restart, which can reduce the restart time. Thus, if you have a large number of autoinstalled terminals, the restart time can be significantly improved when restart delay is set to zero.

Normal shutdown: CICS deletes autoinstalled terminal entries during normal shutdown unless AIRDELAY ≠0 and the terminal has not been deleted. If the restart delay is set to zero, CICS has not cataloged terminal entries when they were autoinstalled, so they are not deleted. This can reduce normal shutdown time.

XRF takeover: The system initialization parameter, AIRDELAY, should not affect XRF takeover. The tracking process still functions as before regardless of the value of the restart delay. Thus, after a takeover, the alternate system still has all the autoinstalled terminal entries. However, if a takeover occurs before the catchup process completes, some of the autoinstalled terminals have to log on to CICS again. The alternate CICS system has to rely on the catalog to complete the catchup process and, if the restart delay is set to zero in the active system, the alternate system is not able to restore the autoinstalled terminal entries that have not been tracked. Those terminals have to log on to the new CICS system, rather than being switched or rebound after takeover.

You have to weigh the risk of having some terminal users log on again because tracking has not completed, against the benefits introduced by setting the restart delay to zero. Because catchup takes only a few minutes, the chance of such a takeover occurring is usually small.

The delete delay parameter (AILDELAY)

The delete delay system initialization parameter lets you control how long an autoinstalled terminal entry remains available after the terminal has logged off. The default value of zero means that the terminal entry is scheduled for deletion as soon as the terminal is logged off. Otherwise, CICS schedules the deletion of the TCTTE as a timer task.

In general, setting the delete delay to a nonzero value can improve the performance of CICS when many autoinstalled terminals are logging on and off during the day. However, this does mean that unused autoinstalled terminal entry storage is not freed for use by other tasks until the delete delay interval has expired. This parameter provides an effective way of defining a terminal whose storage lifetime is somewhere between that of an autoinstalled terminal and a statically defined terminal.

The effect of setting the delete delay to a nonzero value can have different effects depending on the value of the restart delay:

Nonzero restart delay: When the restart delay is nonzero, CICS catalogs autoinstalled terminal entries in the global catalog.

If the delete delay is nonzero as well, CICS retains the terminal entry so that it is re-used when the terminal logs back on. This can eliminate the overhead of:

- Deleting the terminal entry in virtual storage
- An I/O to the catalog and recovery log
- Re-building the terminal entry when the terminal logs on again.

Zero restart delay: When the restart delay is zero, CICS does not catalog autoinstalled terminal entries in the global catalog whatever value is specified for the delete delay.

If the delete delay is nonzero, CICS retains the terminal entry so that it is re-used when the terminal logs back on. This can save the overhead of deleting the terminal entry in virtual storage and the rebuilding of the terminal entry when the terminal logs on again.

Effects

You can control the use of resource by autoinstall processing in three ways:

1. By using the transaction class limit to restrict the number of autoinstall tasks that can concurrently exist (see page 260).
2. By using the CATA and CATD transactions to install and delete autoinstall terminals dynamically. If you have a large number of devices autoinstalled, shutdown can fail due to the MXT system initialization parameter being reached or CICS becoming short on storage. To prevent this possible cause of shutdown failure, you should consider putting the CATD transaction in a class of its own to limit the number of concurrent CATD transactions.
3. By specifying AIQMAX to limit the number of devices that can be queued for autoinstall. This protects against abnormal consumption of virtual storage by the autoinstall process, caused as a result of some other abnormal event.

   If this limit is reached, the AIQMAX system initialization parameter affects the LOGON and BIND processing by CICS. CICS requests VTAM to stop passing LOGON and BIND requests to CICS. VTAM holds such requests until CICS indicates that it can accept further LOGONs and BINDs (this occurs when CICS has processed a queued autoinstall request).

Recommendations

If the autoinstall process is noticeably slowed down by the AIQMAX limit, raise it. If the CICS system shows signs of running out of storage, reduce the AIQMAX limit. If possible, set the AIQMAX system initialization parameter to a value higher than that reached during normal operations.

In a **non-XRF** environment, settings of (restart delay=0) and (delete delay= hhmmss>0) are the most efficient for processor and DASD utilization. However, this efficiency is gained at a cost of virtual storage, because the TCT entries are not deleted until the delay period expires.

A value of zero for both restart delay and delete delay is the best overall setting for many systems from an overall performance and virtual-storage usage point of view.

If restart delay is greater than zero (cataloging active), the performance of autoinstall is significantly affected by the definition of the global catalog (DFHGCD) and the output data set for transient data. The default buffer specifications used by VSAM for the restart data set may not be sufficient in a high activity system.

Because a considerable number of messages are sent to transient data during logon and logoff, the performance of these output destinations should also be taken into consideration.

In an **XRF** environment, a restart delay value of greater than zero should give better performance when catchup of a large number of autoinstalled terminals is necessary.

How monitored

Monitor the autoinstall rate during normal operations by inspecting the autoinstall statistics regularly.

LU6.2 sessions limit

Within the LU6.2 architecture CICS supported a LU6.2 session limit for each region of 46656. Functional enhancements have allowed this limit to extend to 93312 sessions.

No immediate change to performance occurs when the original limit is passed, but the following considerations should be made when having large numbers of sessions connected to a single CICS region.

- CICS startup time will increase if more sessions are installed at initial startup time.
- CICS restart time will increase with the additional number of sessions.
- You should ensure that the global catalogue is large enough to hold the additional number of records.
- You should ensure that the log structures can accommodate the extra load imposed by so many sessions.
- You should distribute the high number of sessions across multiple CICS regions, if possible. From a resource contention point of view this can relieve CPU TCB contention and I/O contention to the catalogue.

- It is also advisable to spread high numbers of sessions across different CICS regions for availability reasons, so that the whole network is not lost if failure occurs.

When a higher number of sessions are installed into a region you can potentially experience higher transaction rates and it is recommended that you undertake a resource capacity planning exercise. A storage capacity planning exercise should be carried out anyway for the increase in storage requirement for any additional sessions

Chapter 18. VSAM and file control

This chapter discusses performance tuning issues related to VSAM and file control.

VSAM considerations: general objectives

Tuning consists of providing a satisfactory level of service from a system at an acceptable cost. A satisfactory service, in the case of VSAM, is likely to be obtained by providing adequate buffers to minimize physical I/O and, at the same time, allowing several operations concurrently on the data sets.

The costs of assigning additional buffers and providing for concurrent operations on data sets are the additional virtual and real storage that is required for the buffers and control blocks.

Several factors influence the performance of VSAM data sets. The rest of this section reviews these and the following sections summarize the various related operands of file control.

Note that, in this section, a distinction is made between "files" and "data sets":

- A "file" means a view of a data set as defined by an FCT file entry and a VSAM ACB.
- A "data set" means a VSAM "sphere", including the base cluster with any associated AIX paths.

If you wish to use existing VSAM file definitions, see the *CICS/ESA Resource Definition Guide*.

Local shared resources (LSR) or Nonshared resources (NSR)

The first decision to make for each file is whether to use LSR or NSR for its VSAM buffers and strings. It is possible to use up to eight separate LSR pools for file control files. There is also a decision to make on how to distribute the data sets across the LSR pools.

CICS provides separate LSR buffer pools for data and index records. If only data buffers are specified, only one set of buffers are built and used for both data and index records.

Note that all FCT files opened for access to a particular VSAM data set normally must use the same resource type: see “Data set name sharing” on page 220.

LSR files share a common pool of buffers and a common pool of strings (that is, control blocks supporting the I/O operations). Other control blocks define the file and are unique to each file or data set. NSR files or data sets have their own set of buffers and control blocks.

Some important differences exist between NSR and LSR in the way that VSAM allocates and shares the buffers.

In NSR, the minimum number of data buffers is STRNO + 1, and the minimum index buffers (for KSDSs and AIX paths) is STRNO. One data and one index buffer are preallocated to each string, and one data buffer is kept in reserve for CI splits. If there are extra data buffers, these are assigned to the first sequential operation; they may also be used to speed VSAM CA splits by permitting chained I/O operations. If there are extra index buffers, they are shared between the strings and are used to hold high-level index records, thus providing an opportunity for saving physical I/O.

In LSR, there is no preallocation of buffers to strings, or to particular files or data sets. When VSAM needs to reuse a buffer, it picks the buffer that has been referenced least recently. Strings are always shared across all data sets.

Before issuing a read to disk when using LSR, VSAM first scans the buffers to check if the control interval it requires is already in storage. If so, it may not have to issue the read. This buffer “lookaside” can reduce I/O significantly.

Another important difference between LSR and NSR is in concurrent access to VSAM CIs. NSR allows multiple copies of a CI in storage; you can have one (but only one) string updating a CI and other strings reading different copies of the same CI. In LSR, there is only one copy of a CI in storage; the second of the requests must queue until the first operation completes. LSR permits several read operations to share access to the same buffer, but updates require exclusive use of the buffer and must queue until a previous update or previous reads have completed; reads must wait for any update to finish. It is possible, therefore, that transactions with concurrent browse and update operations that run successfully with NSR may, with LSR, hit a deadlock as the second operation waits unsuccessfully for the first to complete.

Transactions should always be designed and programmed to avoid deadlocks. For further discussions, see the *CICS/ESA Application Programming Guide*.

LSR has significant advantages, by providing:

- More efficient use of virtual storage because buffers and strings are shared.
- Better performance because of better buffer lookaside, which can reduce I/O operations.
- Self-tuning because more buffers are allocated to busy files and frequently referenced index control intervals are kept in its buffers.
- Better read integrity because there is only one copy of a CI in storage.
- Use of synchronous file requests and a UPAD exit. CA and CI splits for LSR files do not cause either the subtask or main task to wait. VSAM takes the

UPAD exit while waiting for physical I/O, and processing continues for other CICS work during the CA/CI split.

File control requests for NSR files are done asynchronously, however, and still cause the CICS main task or subtask to stop during a split.

NSR, on the other hand:

- Allows for specific tuning in favor of a particular data set
- Can provide better performance for sequential operations.

The general recommendation is to use LSR for all VSAM data sets except where you have one of the following situations:

- A file is very active but there is no opportunity for lookaside because, for instance, the file is very large.
- High performance is required by the allocation of extra index buffers.
- Fast sequential browse or mass insert is required by the allocation of extra data buffers.
- Control area (CA) splits are expected for a file, and extra data buffers are to be allocated to speed up the CA splits.

If you have only one LSR pool, a particular data set cannot be isolated from others using the same pool when it is competing for strings, and it can only be isolated when it is competing for buffers by specifying unique CI sizes. In general, you get more self-tuning effects by running with one large pool, but it is possible to isolate busy files from the remainder or give additional buffers to a group of high performance files by using several pools. It is possible that a highly active file has more successful buffer lookaside and less I/O if it is set up as the only file in an LSR subpool rather than using NSR. Also the use of multiple pools eases the restriction of 255 strings for each pool.

Number of strings

The next decision to be made is the number of concurrent accesses to be supported for each file and for each LSR pool.

This is achieved by specifying VSAM “strings”. A string is a request to a VSAM data set requiring “positioning” within the data set. Each string specified results in a number of VSAM control blocks (including a “placeholder”) being built.

VSAM requires one or more strings for each concurrent file operation. For nonupdate requests (for example, a READ or BROWSE), an access using a base needs one string, and an access using an AIX needs two strings (one to hold position on the AIX and one to hold position on the base data set). For update requests where no upgrade set is involved, a base still needs one string, and a path two strings. For update requests where an upgrade set is involved, a base needs 1+n strings and a path needs 2+n strings, where ‘n’ is the number of members in the upgrade set (VSAM needs one string per upgrade set member to hold position). Note that, for each concurrent request, VSAM can reuse the n strings required for upgrade set processing because the upgrade set is updated serially. See “CICS calculation of LSR pool operands” on page 219.

A simple operation such as read direct frees the string or strings immediately, but a read for update, mass insert, or browse retains them until a corresponding release, update, or end browse is performed.

The interpretation of the STRNO operand by CICS and by VSAM differs depending upon the context:

- The equivalent STRINGS operand of the file definition has the same meaning as the STRNO in the VSAM ACB for NSR files: that is, the actual number of concurrent outstanding VSAM requests that can be handled. When AIX paths or upgrade sets are used, the actual number of strings which VSAM allocates to support this may be greater than the STRNO value specified.
- The equivalent STRINGS operand of the LSRPOOL definition has the same meaning as the STRNO in the VSAM BLDVRP macro: that is, the absolute number of strings to be allocated to the resource pool. Unless an LSR pool contains only base data sets, the number of concurrent requests that can be handled is less than the STRNO value specified.

Note: There are some special considerations for setting the STRINGS value for an ESDS (see “Number of strings considerations for ESDS files” on page 217).

For LSR, it is possible to specify the precise numbers of strings, or to have CICS calculate the numbers. The number specified in the SHRCTL macro is the actual number of strings in the pool. If CICS is left to calculate the number of strings, it derives the pool STRNO from the FCT file entries and interpret this, as with NSR, as the actual number of concurrent requests. (For an explanation of CICS calculation of LSR pool operands, see “CICS calculation of LSR pool operands” on page 219.)

You must decide how many concurrent read, browse, updates, mass inserts, and so on you need to support.

If access to a file is read only with no browsing, there is no need to have a large number of strings; just one may be sufficient. Note that, while a read operation only holds the VSAM string for the duration of the request, it may have to wait for the completion of an update operation on the same CI.

In general (but see “Number of strings considerations for ESDS files” on page 217) where some browsing or updates are used, STRNO should be set to 2 or 3 initially and CICS file statistics should be checked regularly to see the proportion of wait-on-strings encountered. Wait-on-strings of up to 5% of file accesses would usually be considered quite acceptable. You should not try, with NSR files, to keep wait-on-strings permanently zero.

CICS manages string usage for both files and LSR pools. For each file, whether it uses LSR or NSR, CICS limits the number of concurrent VSAM requests to the STRNO= specified in the FCT file entry. For each LSR pool, CICS also prevents more requests being concurrently made to VSAM than can be handled by the strings in the pool. Note that, if additional strings are required for upgrade-set processing at update time, CICS anticipates this requirement by reserving the additional strings at read-for-update time. If there are not enough file or LSR pool strings available, the requesting task waits until they are freed. The CICS statistics give details of the string waits.

When deciding the number of strings for a particular file, consider the maximum number of concurrent tasks. Because CICS command level does not allow more than one request to be outstanding against a particular data set from a particular task, there is no point in allowing strings for more concurrent requests.

If you want to distribute your strings across tasks of different types, the transaction classes may also be useful. You can use transaction class limits to control the transactions issuing the separate types of VSAM request, and for limiting the number of task types that can use VSAM strings, thereby leaving a subset of strings available for other uses.

All placeholder control blocks must contain a field long enough for the largest key associated with any of the data sets sharing the pool. Assigning one inactive file that has a very large key (primary or alternate) into an LSR pool with many strings may use excessive storage.

Number of strings considerations for ESDS files

There are some special performance considerations when choosing a STRINGS valus for an ESDS file.

If an ESDS is used as an 'add-only' file (that is, it is used only in write mode to add records to the end of the file), a string number of 1 is strongly recommended. Any string number greater than 1 can significantly affect performance, because of exclusive control conflicts that occur when more than one task attempts to write to the ESDS at the same time.

If an ESDS is used for both writing and reading, with writing, say, being 80% of the activity, it is better to define two file definitions—using one file for writing and the other for reading.

Size of control intervals

The size of the data set control intervals is not an operand specified to CICS; it is defined through VSAM AMS. However, it can have a significant performance effect on a CICS system that provides access to the control interval.

In general, direct I/O runs slightly more quickly when data CIs are small, whereas sequential I/O is quicker when data CIs are large. However, with NSR files, it is possible to get a good compromise by using small data CIs but also assigning extra buffers, which leads to chained and overlapped sequential I/O. However, all the extra data buffers get assigned to the first string doing sequential I/O.

VSAM functions most efficiently when its control areas are the maximum size, and it is generally best to have data CIs larger than index CIs. Thus, typical CI sizes for data are 4KB to 12KB and, for index, 1KB to 2KB.

In general, you should specify the size of the data CI for a file, but allow VSAM to select the appropriate index CI to match. An exception to this is if key compression turns out to be less efficient than VSAM expects it to be. In this case, VSAM may select too small an index CI size. You may find an unusually high rate of CA splits occurring with poor use of DASD space. If this is suspected, specify a larger index CI.

In the case of LSR, there may be a benefit in standardizing on the CI sizes, because this allows more sharing of buffers between files and thereby allow a lower

total number of buffers. Conversely, there may be a benefit in giving a file unique CI sizes to prevent it from competing for buffers with other files using the same pool.

Try to keep CI sizes at 512, 1KB, 2KB, or any multiple of 4KB. Unusual CI sizes like 26KB or 30KB should be avoided. A CI size of 26KB does not mean that physical block size will be 26KB; the physical block size will most likely be 2KB in this case (it is device-dependent).

Number of buffers (NSR)

The next decision is the number of buffers to be provided for each file. Enough buffers must be provided to support the concurrent accesses specified in the STRNO operand for the file (in fact VSAM enforces this for NSR).

Specify the number of data and index buffers for NSR using the DATABUFFER and INDEXBUFFER operands of the file definition (or explicitly by coding the BUFND and BUFNI operands in the file definition on the CSD). It is important to specify sufficient index buffers. If a KSDS consists of just one control area (and, therefore, just one index CI), the minimum index buffers equal to STRNO is sufficient. But when a KSDS is larger than this, at least one extra index buffer needs to be specified so that at least the top level index buffer is shared by all strings. Further index buffers reduces index I/O to some extent.

BUFND should generally be the minimum at STRNO + 1, unless the aim is to enable overlapped and chained I/O in sequential operations or it is necessary to provide the extra buffers to speed up CA splits.

Note that when the file is an AIX path to a base, the same BUFNI (if the base is a KSDS) and BUFND are used for AIX and base buffers (but see “Data set name sharing” on page 220).

Number of buffers (LSR)

The set of buffers of one size in an LSR pool is called a “subpool.” The number of buffers for each subpool is controlled by the DATA and INDEX operands of the LSRPOOL definition It is possible to specify precise numbers or to have CICS calculate the numbers. (The method used by CICS to calculate the number of buffers is described below.)

Allowing CICS to calculate the LSR operands is easy but it requires additional overhead (at startup) to build the pool because CICS must read the VSAM catalog for every file that is specified to use the pool. Also it cannot be fine-tuned by specifying actual quantities of each buffer size. When making changes to the size of an LSR pool, refer to the CICS shutdown statistics before and after the change is made. These statistics show whether the proportion of VSAM reads satisfied by buffer lookaside is significantly changed or not.

In general, you would expect to benefit more by having extra index buffers for lookaside, and less by having extra data buffers. This is a further reason for standardizing on LSR data and index CI sizes, so that one subpool does not have a mix of index and data CIs in it.

Note: Data and index buffers are specified separately with the LSRPOOL definition. Thus, there is not a requirement to use CI size to differentiate between data and index values.

Take care to include buffers of the right size. If no buffers of the required size are present, VSAM uses the next larger buffer size.

CICS calculation of LSR pool operands

If you have not specified LSR operands for a pool, CICS calculates for you the buffers and strings required. To do this, it scans the FCT for all file entries for files specified to use the pool. For each, it uses:

- From the FCT file entries:
  - The number of strings, STRNO=.
- From the VSAM catalog:
  - The levels of index for each of these files
  - The CI sizes
  - The keylengths for the base, the path (if it is accessed through an AIX path), and upgrade set AIXs.

Note: If you have specified only buffers or only strings, CICS performs the calculation for what you have not specified.

The following information helps you calculate the buffers required. A particular file may require more than one buffer size. For each file, CICS determines the buffer sizes required for:

- The data component
- The index component (if a KSDS)
- The data and index components for the AIX (if it is an AIX path)
- The data and index components for each AIX in the upgrade set (if any).

The number of buffers for each is calculated as follows:

- For data components (base and AIX) = (STRNO= in the FCT entry) + 1
- For index components (base and AIX) = (STRNO= in the FCT entry) + (the number of levels in the index) – 1
- For data and index components for each AIX in the upgrade set, one buffer each.

When this has been done for all the files to use the pool, the total number of buffers for each size is:

- Reduced to either 50% or the percentage specified in the RSCLMT operand to the SHRCTL macro. The RSCLMT operand takes precedence.
- If necessary, increased to a minimum of three buffers.
- Rounded up to the nearest 4KB boundary.

To calculate the number of strings, CICS determines the number of strings to handle concurrent requests for each file as the sum of:

- STRNO= strings for the base
- STRNO= strings for the AIX (if it is an AIX path)
- n strings if there is an upgrade set (where ‘n’ is the number of members in the upgrade set).

Note: If the LSRPOOL is calculated by CICS and the data sets have been archived by HSM, the startup time of a CICS system can be considerably lengthened because the data sets are needed one by one. CICS obtains the necessary catalog information, but it does not open the database. Therefore the database is still effectively archived. This problem recurs when the system is started again, and remains until the data set has been opened.

When the strings have been accumulated for all files, the total is:

- Reduced to either 50% or the percentage specified in the RSCLMT operand to the SHRCTL macro. The RSCLMT operand takes precedence.
- Reduced to 255 (the maximum number of strings allowed for a pool by VSAM).
- Increased to the largest specified STRNO= for a particular file.

The operands calculated by CICS are shown in the CICS statistics.

Switching data sets from RLS mode to LSR mode

Although it is not generally recommended, there may be occasions when you need to switch a data set from RLS mode to non-RLS mode (for example, to read-only LSR mode during a batch update). This could lead to the LSR pools that are not explicitly defined, and which CICS builds using default values, not having suffiicent resources to support files switched to LSR mode after the pool has been built.

To avoid files failing to open because of the lack of adequate resources, you can specify that CICS should include files opened in RLS mode when it is calculating the size of an LSR pool using default values. To specify the inclusion of files defined with RLSACCESS(YES) in an LSR pool being built using values that CICS calculates, use the RLSTOLSR=YES system initialization parameter (RLSTOLSR=NO is the default)

See the *CICS/ESA System Definition Guide* for more information about the RLSTOLSR parameter.

Data set name sharing

Data set name (DSN) sharing (MACRF=DSN specified in the VSAM ACB) is the default for all VSAM data sets. It causes VSAM to create a single control block structure for the strings and buffers required by all the files that relate to the same base data set cluster, whether as a path or direct to the base. VSAM makes the connection at open time of the second and subsequent files. Only if DSN sharing is specified VSAM realizes that it is processing the same data set.

This single structure:

- Provides VSAM update integrity for multiple ACBs (FCT file entries) updating one VSAM data set
- Allows the use of VSAM share options 1 or 2, while still permitting multiple update ACBs within the CICS region
- Saves virtual storage.

DSN sharing is the default for files using both NSR and LSR. The only exception to this default is made when opening a file that has been specified as read-only (SERVREQ=READ or SERVREQ=BROWSE) and with DSNSHR=UPDATE in the FCT. CICS provides this option so that a file (represented by an FCT file entry)

can be isolated from other uses of that same data set in a different LSR pool or in NSR by suppressing DSN sharing. CICS ignores this operand for files with update, add, or delete options because VSAM would not then be able to provide update integrity if two file control file entries were updating the same data set concurrently.

The BASE= operand is associated with DSN sharing. It is used to group together FCT entries that are to refer to the same VSAM base data set. BASE=name has no effect for data sets that use LSR.

When the first member of a group of DSN-sharing NSR files is opened, CICS must specify to VSAM the total number of strings to be allocated for all file entries in the group, by means of the BSTRNO value in the ACB. VSAM builds its control block structure at this time regardless of whether the first data set to be opened is a path or a base. CICS calculates the value of BSTRNO used at the time of the open as part of the assembly of the FCT by adding the STRNO values in all the FCT entries that share the same BASE= operand.

If you do not provide the BASE= operand, the VSAM control block structure may be built with insufficient strings for later processing. This should be avoided for performance reasons. In such a case, VSAM invokes the dynamic string addition feature to provide the extra control blocks for the strings as they are required, and the extra storage is not released until the end of the CICS run.

AIX considerations

For each AIX defined with the UPGRADE attribute, VSAM upgrades the AIX automatically when the base cluster is updated.

For NSR, VSAM uses a special set of buffers associated with the base cluster to do this. This set consists of two data buffers and one index buffer, which are used serially for each AIX associated with a base cluster. It is not possible to tune this part of the VSAM operation.

For LSR, VSAM uses buffers from the appropriate subpool.

Care should be taken when specifying to VSAM that an AIX should be in the upgrade set. Whenever a new record is added, an existing record deleted, or a record updated with a changed attribute key, VSAM updates the AIXs in the upgrade set. This involves extra processing and extra I/O operations.

Situations that cause extra physical I/O

Listed below are some situations that can lead to a lot of physical I/O operations, thus affecting both response times and associated processor pathlengths:

- When a KSDS is defined with SHROPT of 4, all direct reads cause a refresh of both index and data buffers (to ensure latest copy).
- Any sequence leading to CICS issuing ENDREQ invalidates all data buffers associated with the operation. This may occur when you end a get-update (without the following update), a browse (even a start browse with a no-record-found response), a mass-insert or any get-locate from a program. If the operation is not explicitly ended by the program, CICS ends the operation at syncpoint or end of task.

- If there are more data buffers than strings, a start browse causes at least half the buffers to participate immediately in chained I/O. If the browse is short, the additional I/O is unnecessary.

Other VSAM definition operands

Free space operands need to be selected with care, and can help reduce the number of CI and CA splits. Where records are inserted all over a VSAM data set, it is appropriate to include free space in each CI. Where the inserts are clumped, free space in each CA is required. If all the inserts take place at just a few positions in the file, VSAM should be allowed to split the CA, and it is not necessary to specify any free space at all.

Adding records to the end of a VSAM data set does ***not*** cause CI/CA splits. Adding sequential records to anywhere but the end causes splits. An empty file with a low-value dummy key tends to reduce splits; a high-value key increases the number of splits.

VSAM resource usage (LSRPOOL)

The default for all VSAM data sets is LSR. If multiple pools are supported CICS provides for the use of pools 1 through 8

Effects

The LSRPOOL operand specifies whether a file is to use LSR or NSR and, if LSR, which pool.

Where useful

The LSRPOOL operand can be used in CICS systems with VSAM data sets.

Limitations

All files with the same base data set, except read-only files with DSNSHR=UPDATE specified in the FCT file entry, must use either the same LSR pool or all use NSR.

SERVREQ=REUSE files cannot use LSR.

Recommendations

See “VSAM considerations: general objectives” on page 213. Consider removing files from an LSR pool.

How implemented

The resource usage is defined by the LSRPOOL definition on the CSD. For more information about the CSD, see the *CICS/ESA Resource Definition Guide*.

VSAM buffer allocations for NSR (BUFNI and BUFND)

For files using nonshared resources (NSR), the BUFNI and BUFND operands define VSAM index buffers and data buffers respectively.

Effects

BUFNI and BUFND specify the number of index and data buffers for an NSR file.

The number of buffers can have a significant effect on performance. The use of many buffers can permit multiple concurrent operations (if there are the corresponding number of VSAM strings) and efficient sequential operations and CA splits. Providing extra buffers for high-level index records can reduce physical I/O operations.

Buffer allocations above the 16MB line represent a significant part of the virtual storage requirement of most CICS systems.

Further information on BUFNI and BUFND is given in the VSAM publications, for example, the *MVS VSAM Operation* manual, SC31-6408.

BUFNI and BUFND have no effect if they are specified for files using LSR.

Where useful

The BUFND and BUFNI operands should be used in CICS systems that use VSAM NSR files in CICS file control.

Limitations

These operands can be overridden by VSAM if they are insufficient for the strings specified for the VSAM data set. The maximum specification is 255. A specification greater than this will automatically be reduced to 255. Overriding of VSAM strings and buffers should never be done by specifying the AMP= attribute on the DD statement.

Recommendations

See “VSAM considerations: general objectives” on page 213.

How implemented

The BUFNI and BUFND operands are defined in the file definition on the CSD. They correspond exactly to VSAM ACB operands: BUFNI is the number of index buffers, BUFND is the number of data buffers.

For LSR files, they are ignored.

How monitored

The effects of these operands can be monitored through transaction response times and data set and paging I/O rates. The CICS file statistics show data set activity to VSAM data sets. The VSAM catalog and RMF can show data set activity, I/O contention, space usage, and CI size.

CICS monitoring exception records are produced when a wait for string is necessary, and can be very useful in identifying files that may need more strings allocated.

VSAM buffer allocations for LSR (BUFFERS)

For files using shared resources (LSR), the number of buffers to be used is not specified explicitly by file. The files share the buffers of the appropriate sizes in the LSR pool. The number of buffers in the pool may either be specified explicitly using the BUFFERS operand in the file definition on the CSD, or be left to CICS to calculate. For more information about the CSD, see the *CICS/ESA Resource Definition Guide*.

Effects

The BUFFERS operand allows for exact definition of specific buffers for the LSR pool.

The number of buffers can have a significant effect on performance. The use of many buffers can permit multiple concurrent operations (if there are the corresponding number of VSAM strings). It can also increase the chance of successful buffer lookaside with the resulting reduction in physical I/O operations.

The number of buffers should achieve an optimum between increasing the I/O saving due to lookaside and increasing the real storage requirement. This optimum is different for buffers used for indexes and buffers used for data. Note that the optimum buffer allocation for LSR is likely to be significantly less than the buffer allocation for the same files using NSR.

Where useful

The BUFFERS operand should be used in CICS systems that use VSAM LSR files in CICS file control.

Recommendations

See “VSAM considerations: general objectives” on page 213.

How implemented

The BUFFERS operand is defined in the file definition on the CSD. For more information about the CSD, see the *CICS/ESA Resource Definition Guide*.

How monitored

The effects of these operands can be monitored through transaction response times and data set and paging I/O rates. The effectiveness affects both file and lsrpool statistics. The CICS file statistics show data set activity to VSAM data sets. The VSAM catalog and RMF can show data set activity, I/O contention, space usage, and CI size.

VSAM string settings for NSR (STRNO)

STRNO is used to determine the number of concurrent operations possible against the file and against the VSAM base cluster to which the file relates.

Effects

The STRNO operand for files using NSR has the following effects:

- It specifies the number of concurrent asynchronous requests that can be made against that specific file.
- It is used as the STRNO in the VSAM ACB.
- It is used, in conjunction with the BASE operand, to calculate the VSAM BSTRNO.
- A number greater than 1 can adversely affect performance for ESDS files used exclusively in write mode. With a string number greater than 1, the cost of invalidating the buffers for each of the strings is greater than waiting for the string, and there can be a significant increase in the number of VSAM EXCP requests.

Strings represent a significant part of the virtual storage requirement of most CICS systems. With CICS, this storage is above the 16MB line.

Where useful

The STRNO operand should be used in CICS systems that use VSAM NSR files in CICS file control.

Limitations

A maximum of 255 strings can be used as the STRNO or BSTRNO in the ACB.

Recommendations

See “Number of strings considerations for ESDS files” on page 217 and “VSAM considerations: general objectives” on page 213.

How implemented

The number of strings is defined by the STRINGS operand in the CICS file definition on the CSD. It corresponds to the VSAM operand in the ACB except where a base file is opened as the first for a VSAM data set; in this case, the CICS-accumulated BSTRNO value is used as the STRNO for the ACB.

How monitored

The effects of the STRNO operand can be seen in increased response times and monitored by the string queueing statistics for each FCT entry. RMF can show I/O contention in the DASD subsystem.

VSAM string settings for LSR (STRNO)

STRNO is used to determine the number of strings and thereby the number of concurrent operations possible against the LSR pool (assuming that there are buffers available).

Effects

The STRNO operand relating to files using LSR has the following effects:

- It specifies the number of concurrent requests that can be made against that specific file.
- It is used by CICS to calculate the number of strings and buffers for the LSR pool.
- It is used as the STRNO for the VSAM LSR pool.
- It is used by CICS to limit requests to the pools to prevent a VSAM short-on-strings condition (note that CICS calculates the number of strings required per request).
- A number greater than 1 can adversely affect performance for ESDS files used exclusively in write mode. With a string number greater than 1, the cost of resolving exclusive control conflicts is greater than waiting for a string. Each time exclusive control is returned, a GETMAIN is issued for a message area, followed by a second call to VSAM to obtain the owner of the control interval.

Where useful

The STRNO operand can be used in CICS systems with VSAM data sets.

Limitations

A maximum of 255 strings is allowed per pool.

Recommendations

See “Number of strings considerations for ESDS files” on page 217 and “VSAM considerations: general objectives” on page 213.

How implemented

The number of strings is defined by the STRNO operand in the file definition on the CSD, which limits the concurrent activity for that particular file.

How monitored

The effects of the STRNO operand can be seen in increased response times for each file entry. The CICS LSRPOOL statistics give information on the number of data set accesses and the highest number of requests for a string.

Examination of the string numbers in the CICS statistics shows that there is a two-level check on string numbers available: one at the data set level (see “Transaction statistics” on page 425), and one at the shared resource pool level (see “Transaction initialization” on page 522).

RMF can show I/O contention in the DASD subsystem.

Maximum keylength for LSR (KEYLEN)

The KEYLEN operand in the file definition on the CSD specifies the size of the largest key to be used in an LSR pool.

The maximum keylength may be specified explicitly using the KEYLEN operand in the file definition on the CSD, or it may be left to CICS to determine from the VSAM catalog. For more information about the CSD, see the *CICS/ESA Resource Definition Guide*.

Effects

The KEYLEN operand causes the "placeholder" control blocks to be built with space for the largest key that can be used with the LSR pool. If the KEYLEN specified is too small, it prevents requests for files that have a longer key length.

Where useful

The KEYLEN operand can be used in CICS systems with VSAM data sets.

Recommendations

See "VSAM considerations: general objectives" on page 213.

The key length should always be as large as, or larger than, the largest key for files using the LSR pool.

How implemented

The size of the maximum keylength is defined in the KEYLEN operand in the file definition on the CSD. For more information about the CSD, see the *CICS/ESA Resource Definition Guide*.

Resource percentile for LSR (RSCLMT)

The RSCLMT operand in the file definition on the CSD specifies the percentage of the buffers and strings that CICS should apply to the value that it calculates.

Effects

The method used by CICS to calculate LSR pool operands and the use of the RSCLMT value is described in "VSAM considerations: general objectives" on page 213.

This operand has no effect if both the BUFFERS and the STRNO operands are specified for the pool.

Where useful

The RSCLMT operand can be used in CICS systems with VSAM data sets.

Recommendations

See “VSAM considerations: general objectives” on page 213.

Because RSCLMT can be applied only to files that are allocated at initialization of the LSR pool (when the first file in the pool is opened), it is always wise to specify the decimal STRNO and BUFFERS for an LSR pool.

How implemented

The RSCLMT operand is specified in the file definition on the CSD. For more information about the CSD, see the *CICS/ESA Resource Definition Guide*.

VSAM local shared resources (LSR)

Effects

CICS always builds a control block for LSR pool 1. CICS builds control blocks for other pools if either a SHRCTL macro is included in the FCT, or a file in the FCT (at CICS initialization time) has LSRPOOL= defined with the number of the pool. If DFHFCT is currently being used to create VSAM files, data tables, or LSR pools, they must be migrated to the CICS system definition data set (CSD). For more information about the CSD, see the *CICS/ESA Resource Definition Guide*.

Where useful

VSAM shared resources can be used in CICS systems that use VSAM.

Recommendations

See “VSAM considerations: general objectives” on page 213.

How implemented

CICS uses the operands provided in the SHRCTL macro to build the LSR pool.

How monitored

VSAM LSR can be monitored by means of response times, paging rates, and CICS LSRPOOL statistics. The CICS LSRPOOL statistics show string usage, data set activity, and buffer lookasides (see “Transaction initialization” on page 522).

Hiperspace buffers

VSAM Hiperspace buffers reside in MVS expanded storage. These buffers are backed only by expanded storage. If the system determines that a particular page of this expanded storage is to be used for another purpose, the current page’s contents are discarded rather than paged-out. If VSAM subsequently requires this page, it retrieves the data from DASD. VSAM manages the transfer of data between its Hiperspace buffers and its CICS address space buffers. CICS file control can only work with VSAM data when it is in a CICS address space buffer. Data is transferred between Hiperspace buffers and address space buffers in blocks of pages using CREAD and CWRITE commands. See “Transaction termination” on page 523 for more information.

Effects

The use of a very large number of Hiperspace buffers can reduce both physical I/O and pathlength when accessing your CICS files because the chance of finding the required records already in storage is relatively high.

Limitations

Because the amount of expanded storage is limited, it is possible that the installation will overcommit its use and VSAM may be unable to allocate all of the Hiperspace buffers requested. MVS may use expanded storage pages for purposes other than those allocated to VSAM Hiperspace buffers. In this case CICS continues processing using whatever buffers are available.

If address space buffers are similarly overallocated then the system would have to page. This overallocation of address space buffers is likely to seriously degrade CICS performance whereas overallocation of Hiperspace buffers is not.

Hiperspace buffer contents are lost when an address space is swapped out. This causes increased I/O activity when the address is swapped in again. If you use Hiperspace buffers, you should consider making the CICS address space nonswappable.

Recommendations

Keeping data in memory is usually very effective in reducing the CPU costs provided adequate central and expanded storage is available. Using mostly Hiperspace rather than all address space buffers can be the most effective option especially in environments where there are more pressing demands for central storage than VSAM data.

How implemented

CICS never requests Hiperspace buffers as a result of its own resource calculations. You have to specify the size and number of virtual buffers and Hiperspace buffers that you need.

You can use the RDO operands of HSDATA and HSINDEX, which are added to the LSRPOOL definition to specify Hiperspace buffers. Using this method you can adjust the balance between Hiperspace buffers and virtual buffers for your system.

For further details of the CEDA transaction, see the *CICS/ESA Resource Definition Guide*.

Subtasking: VSAM (SUBTSKS=1)

CICS always has three TCBs for normal processing.

- The quasi-reentrant TCB executes the quasi-reentrant application code and most CICS code.
- The resource-owning TCB is used for program loading and security.
- The file-owning TCB is used for opening and closing data sets.

There are two other TCBs:

- The concurrent TCB, which is responsible for VSAM subtasking. This TCB is only active if SUBTSKS=1 is specified.
- The fifth TCB is described in the *CICS/ESA Front End Programming Interface User's Guide*. This TCB is active only if FEPI=YES is specified.

Effects

The objective of subtasks is to increase the maximum throughput of a single CICS system on multiprocessors. However, the intertask communication increases total processor utilization.

When I/O is done on subtasks, any extended response time which would cause the CICS region to stop, such as CI/CA splitting in NSR pools, causes only the additional TCB to stop. This may allow more throughput in a region that has very many CA splits in its file, but has to be assessed cautiously with regard to the extra overhead associated with using the subtask.

APAR PQ09962

MJO 7/7/98

When the SUBTASKS=1 system initialization parameter has been specified:

- All VSAM file control WRITE requests to KSDS are subtasked.
- All other file control requests are never subtasked.
- Auxiliary temporary storage or intrapartition transient data requests are subtasked.
- Resource security checking requests are subtasked when the CICS main TCB (quasi-reentrant mode) exceeds approximately 70% activity.

Where useful

Subtasking can be useful with CICS systems that use VSAM.

Subtasking should only be used in a multiprocessing system in a region that is limited by a single processor but has spare capacity on other processors in the MVS image. If used in other circumstances, it can cause throughput degradation because of the dispatching of multiple tasks.

Limitations

Subtasking can improve throughput only in multiprocessor MVS images, because additional processor cycles are required to run the extra subtask. For that reason, we do not recommend the use of this facility on uniprocessors (UPs). It should be used only for a region that reaches the maximum capacity of one processor in a complex that has spare processor capacity or has NSR files that undergo frequent CI/CA splitting.

Regions that do not contain significant amounts of VSAM data set activity (particularly update activity) do not gain from VSAM subtasking.

Application task elapsed time may increase or decrease because of conflict between subtasking overheads and better use of multiprocessors. Task-related DSA occupancy increases or decreases proportionately.

Recommendations

SUBTSKS=1 should normally be specified only when the CICS system is run on a MVS image with two or more processors ***and*** the peak processor utilization due to the CICS main TCB in a region exceeds, say, about 70% of one processor, and a significant amount of I/O activity within the CICS address space is eligible for subtasking.

In this environment, the capacity of a second processor can be utilized to perform the I/O scheduling activity for VSAM data sets, auxiliary temporary storage, and intrapartition transient data.

The maximum system throughput of this sort of CICS region can be increased by using the I/O subtask, but at the expense of some additional processing for communication between the subtask and the MVS task under which the transaction processing is performed. This additional processing is seldom justified unless the CICS region has reached or is approaching its throughput limit.

A TOR that is largely or exclusively routing transactions to one or more AORs has very little I/O that is eligible for subtasking. It is not, therefore, a good candidate for subtasking.

An AOR is a good candidate only if a significant amount of VSAM I/O is performed within the AOR rather than being function-shipped to an FOR.

Subtasking should be considered for a busy FOR that often has a significant amount of VSAM I/O (but remember that DL/I processing of VSAM data sets is ***not*** subtasked).

How implemented

The system initialization parameter, SUBTSKS=1, defines that subtasking is to be used.

How monitored

CICS dispatcher domain statistics include information about the following:

- The quasi-reentrant mode (QR) TCB
- The resource-owning mode (RO) TCB
- The file-owning mode (FO) TCB
- The concurrent mode TCB (CO) (if SUBTSKS=1 is specified)
- The FEPI (SZ) TCB (if FEPI=YES is specified).

Note: CMF data and CICS trace are fully available.

Data tables

Data tables enable you to build, maintain and have rapid access to data records contained in tables held in virtual storage above the 16MB line. Therefore, they can provide a substantial performance benefit by reducing DASD I/O and pathlength resources. The pathlength to retrieve a record from a data table is significantly shorter than that to retrieve a record already in a VSAM buffer.

Effects

- After the initial data table load operation, DASD I/O can be eliminated for all user-maintained and for read-only CICS-maintained data tables.
- Reductions in DASD I/O for CICS-maintained data tables are dependent on the READ/WRITE ratio. This is a ratio of the number of READs to WRITEs that was experienced on the source data set, prior to the data table implementation. They also depend on the data table READ-hit ratio, that is, the number of READs that are satisfied by the table, compared with the number of requests that go against the source data set.
- CICS file control processor consumption can be reduced by up to 70%. This is dependent on the file design and activity, and is given here as a general guideline only. Actual results vary from installation to installation.

For CICS-maintained data tables, CICS ensures the synchronization of source data set and data table changes. When a file is recoverable, the necessary synchronization is already effected by the existing record locking. When the file is nonrecoverable, there is no CICS record locking and the note string position (NSP) mechanism is used instead for all update requests. This may have a small performance impact of additional VSAM ENDREQ requests in some instances.

Recommendations

- Remember that data tables are defined by two RDO operands, TABLE and MAXNUMRECS of the file definition. No other changes are required.
- Start off gradually by selecting only one or two candidates. You may want to start with a CICS-maintained data table because this simplifies recovery considerations.
- Select a CICS-maintained data table with a high READ to WRITE ratio. This information can be found in the CICS LSRPOOL statistics (see page 522) by running a VSAM LISTCAT job.
- READ INTO is recommended, because READ SET incurs slightly more internal overhead.
- Monitor your real storage consumption. If your system is already real-storage constrained, having large data tables could increase your page-in rates. This in turn could adversely effect CICS system performance. Use your normal performance tools such as RMF to look at real storage and paging rates.
- Remember to select files that have a high proportion of full keyed direct reads as CICS-maintained data table candidates.
- Files that have a large proportion of update activity that does not require to be recovered across a restart would be better suited for user-maintained data tables.
- User-maintained data tables can use the global user exit XDTRD to modify as well as select records. This could allow the user-maintained data table to contain only the information relevant to the application.
- If storage isolation is specified allow for the extra storage needed by the data tables to prevent CICS incurring increased paging.

How implemented

Data tables can be defined using either the CEDA DEFINE FILE panel or DFHFCT TYPE=CICSTABLE or USERTABLE macros. See the *CICS/ESA Resource Definition Guide* for further information.

How monitored

Performance statistics are gathered to assess the effectiveness of the data table. They are in addition to those available through the standard CICS file statistics.

The following information is recorded:

- The number of attempts to read from the table
- The number of unsuccessful read attempts
- The number of bytes allocated to the data table
- The number of records loaded into the data table
- The number of attempts to add to the table
- The number of records rejected by a user exit when being added to the table either during loading or via the API
- The number of attempts to add a record which failed due to the table being full (already at its maximum number of records)
- The number of attempts to update table records via rewrite requests.
- The number of attempts to delete records from the table
- The highest value which the number of records in the table has reached since it was last opened.

There are circumstances in which apparent discrepancies in the statistics may be seen, caused, for example, by the existence of inflight updates.

Chapter 19. Database management

This chapter includes the following topics:

DBCTL minimum threads (MINTHRD)

This parameter specifies the number of threads that are created when CICS connects to DBCTL. They remain allocated while the database resource adapter (DRA) is active, These threads continue to remain allocated until the CICS system is disconnected from DBCTL, unless a thread is stopped by a /STOP command or by a thread failure.

Effects

The DRA allocates control blocks for the specified number of threads at DBCTL connection time. One thread is equivalent to one MVS TCB, thus giving more concurrency on multiprocessors. Because these threads are available for the duration of the DBCTL connection, there is no pathlength overhead for collapsing and reallocating thread related storage, and throughput should, therefore, be faster.

The number you specify should be large enough to cover average DL/I transaction loads. After the MINTHRD limit is reached, additional threads are allocated up to the MAXTHRD limit, the number specified in the MAXREGN, or the maximum of 255, whichever is the lowest.

When multiple CICS systems or Batch message processing programs (BMPs) are connected to DBCTL, the sum of MINTHRD and BMPs must be less than or equal to MAXREGN (MAXREGN is specified in the IMS sysgen macros).

Where useful

MINTHRD can be used in DBCTL systems to synchronize thread allocation with workload requirements.

Limitations

There is currently a storage allocation of about 9KB per thread in the local system queue area (LSQA) below the 16MB line.

Implementation

The MINTHRD and MAXTHRD parameters are specified in the DRA startup table (DFSPZP).

How monitored

DBCTL statistics are available when the CICS/DBCTL interface is shut down normally. The MINTHRD value is recorded (see page 337 for further information). You can also use CICS auxiliary trace to check for queueing for threads and PSBs.

DBCTL maximum threads (MAXTHRD)

The MAXTHRD parameter specifies the maximum number of threads that this CICS system may use up to a value of 255, or the limit imposed by MAXREGN. The default is 1 or the number defined by MINTHRD, whichever is the lowest.

Effects

This parameter controls the maximum number of tasks for which this CICS system can have PSBs scheduled in DBCTL. Any requests to schedule a PSB when the MAXTHRD limit is reached is queued by the DRA.

When multiple CICS systems or Batch message processing programs (BMPs) are connected to DBCTL, the sum of MINTHRD and BMPs must be less than or equal to MAXREGN.

Where useful

MAXTHRD can be used in DBCTL systems to ensure that, at peak loads, additional threads can be built in addition to those already allocated as a result of MINTHRD, thus avoiding waiting for threads.

Limitations

After the MINTHRD limit is exceeded, threads continue to be built up to the MAXTHRD limit but, because each thread's control blocks are allocated during PSB scheduling, the pathlength is greater for the tasks running after the MINTHRD limit has been reached.

Implementation

The MINTHRD and MAXTHRD parameters are specified in the DRA startup table (DFSPZP).

How monitored

DBCTL statistics are available when the CICS/DBCTL interface is shut down normally. The MAXTHRD value is recorded (see page 337 for further information). You can also use CICS auxiliary trace to check for queueing for threads and PSBs.

DBCTL DEDB parameters (CNBA, FPBUF, FPBOF)

Because DEDB parameters are defined both in the CICS region and the IMS/ESA (DBCTL) region, both sets of interdependent parameters are included here.

If you use DEDBs, you must define the characteristics and usage of the IMS/ESA DEDB buffer pool. You do this by specifying parameters (including DRA startup parameters) during IMS/ESA system definition or execution.

The main concerns in defining DEDB buffer pools are the total number of buffers in the IMS/ESA region, and how they are shared by CICS threads. You use the following IMS/ESA FPCTRL parameters to define the number of buffers:

- DBBF: total number of buffers
- DBFX: number of buffers used exclusively by the DEDB system.

The number remaining when you subtract the value specified for DBFX from the value specified for DBBF is the number of buffers available for the needs of CICS threads. In this discussion, we have assumed a fixed number for DBFX. DBBF must, therefore, be large enough to accommodate all batch message processing programs (BMPs) and CICS systems that you want to connect to this DBCTL system.

When a CICS thread connects to IMS/ESA, its DEDB buffer requirements are specified using a normal buffer allocation (NBA) parameter. For a CICS system, there are two NBA parameters in the DRA startup table:

1. CNBA buffers needed for the CICS system. This is taken from the total specified in DBBF.
2. FPBUF buffers to be given to each CICS thread. This is taken from the number specified in CNBA. FPBUF is used for each thread that requests DEDB resources, and so should be large enough to handle the requirements of any application that can run in the CICS system.

A CICS system may fail to connect to DBCTL if its CNBA value is more than that available from DBBF. An application may receive schedule failure if the FPBUF value is more than that available from CNBA. The FPBUF value is used when an application tries to schedule a PSB that contains DEDBs.

When a CICS system has connected to DBCTL successfully, and the application has successfully scheduled a PSB containing DEDBs, the DRA startup parameter FPBOF becomes relevant. FPBOF specifies the number of overflow buffers each thread gets if it exceeds FPBUF. These buffers are not taken from CNBA. Instead, they are buffers that are **serially** shared by all CICS applications or other dependent regions that are currently exceeding their normal buffer allocation (NBA) allocation.

Because overflow buffer allocation (OBA) usage is serialized, thread performance can be affected by NBA and OBA specifications. If FPBUF is too small, more applications need to use OBA, which may cause delays due to contention. If both NBA and OBA are too small, the application fails. If FPBUF is too large, this affects the number of threads that can concurrently access DEDB resources, and increase the number of schedule failures.

Where useful

The DBCTL DEDB parameters are useful in tuning a CICS/DBCTL DEDB fastpath environment.

Recommendations

In a CICS/DBCTL environment, the main performance concern is the trade-off between speed and concurrency. The size of this trade-off is dictated by the kind of applications you are running in the CICS system.

If the applications have approximately the same NBA requirements, there is no trade-off. You can specify an FPBUF large enough to never need OBA. This speeds up access and there is no waste of buffers in CNBA, thus enabling a larger number of concurrent threads using DEDBs.

The more the buffer requirements of your applications vary, the greater the trade-off. If you want to maintain speed of access (because OBAs are not being used) but decrease concurrency, you should increase the value of FPBUF. If you prefer to maintain concurrency, do not increase the value of FPBUF. However, speed of access decreases because this and possibly other threads may need to use the OBA function.

For further guidance on DEDB buffer specification and tuning, see the information on DEDBs in the *IMS/ESA Database Administration Guide*, and the *IMS/ESA System Administration Guide*.

How implemented

DBBF and DBFX are parameters defined during DBCTL system generation or at DBCTL initialization. CNBA, FPBUF, and FPBOF are defined in the DRA startup table (DFSPZP).

How monitored

Monitoring data at the transaction level is returned to CICS by DBCTL at schedule end and transaction termination. This data includes information on DEDB statistics.

Note: To obtain the monitoring data, two event monitoring points (EMPs) must be added to your CICS monitoring control table (MCT). For information about coding the DBCTL EMPs, see the *CICS/ESA Customization Guide*.

DL/I threads (DLTHRED)

The DLTHRED system initialization parameter specifies the number of concurrent DL/I threads that can be allocated for CICS systems with local DL/I, and limits the number of tasks concurrently scheduled for use of IMS/ESA resources. For each DL/I thread, the CICS interface scheduling block (ISB) and the IMS/ESA parameter block (PXPARMS) and partition specification table (PST) control blocks are allocated.

Effects

DLTHRED is used at IMS/ESA DB initialization to create as many sets of control blocks (for example, PSTs) as are required (one set per DL/I thread). The amount of storage required for each DLTHRED varies with the release of IMS:

IMS/ESA 3.1–4.1 12KB

Storage for these control blocks is built at CICS initialization, and remains while CICS is running. A thread is acquired at PSB schedule time, and is held until the PSB or task is terminated or until a syncpoint is reached. In general, the number of threads you specify should be high enough for only 5 to 10% of all tasks to have to wait for threads.

Where useful

The DLTHRED system initialization parameter is used in CICS systems with local DL/I databases.

Limitations

An increase in DLTHRED increases real and virtual storage requirements. If the DLTHRED specification is too low, however, requirements for storage and other processor resources can increase because of frequent task waits for threads, with a consequent increase in task lengths.

Recommendations

There are no special recommendations for DLTHRED.

How implemented

The DLTHRED system initialization parameter is used to set this value and cannot be changed by the CEMT master transaction.

How monitored

CICS monitoring EMPs provide information via Clock 1 on PSB schedule time. This can be an indirect measurement of whether enough threads are allocated. An analysis of the control fields in the DFHDLP module in a dump can show current and maximum used threads.

Also monitor general IMS/ESA DB task throughput and IMS/ESA DB I/O rates. Use auxiliary trace and CICS DL/I statistics (see page 342) to check for queuing for threads, and PSBs.

IMS storage pools (PSBPL, DMBPL, ENQPL)

Three storage pools for use with DL/I control blocks are allocated at CICS initialization by means of system initialization parameters. See *CICS/ESA System Definition Guide* for further information. They are:

- The program specification block pool (PSBPL)
- The data management block pool (DMBPL)
- The enqueue pool (ENQPL).

The PSBPL, DMBPL, and ENQPL system initialization parameters control the use of dynamic storage for DL/I activity in CICS systems with local DL/I databases.

PSBPL and DMBPL establish an upper storage limit for these two pools (PSB and DMB storage respectively), and storage is acquired only when needed. It is not subsequently released.

Effects

The program specification block pool and the data management block pool are both stored in the CICS dynamic storage area (CDSA), while the enqueue pool storage resides in MVS storage, above the 16MB line. All three pool specifications are maximum values and, if that amount of storage is not required for the control blocks, it can be used for other purposes.

A general rule for deciding the initial allocation of PSBPL is to take the largest PSB in the system and multiply by the DLTHRED specification. This ensures that the maximum number of calls can be processed at any one time.

If ENQPL is too small, the IMS DB task abends with a U0775 pseudoabend message, causing dynamic backout of the changes. Generally, this pool is fairly small (2KB to 3KB) but, if shared database is used with batch regions, a larger pool may be required because batch programs probably run for a considerably longer period than online transactions.

If the DMBPL system initialization parameter is too small, the system closes existing unused databases and then opens the required databases.

The enqueue pool (ENQPL) is used with program isolation scheduling, and should be allocated accordingly. With CICS and IMS in a data-sharing environment, the IMS resource lock manager (IRLM) replaces the program isolation scheduling function and the need to specify ENQPL within the CICS region. If IRLM is present in the system, it is automatically used instead of program isolation scheduling.

Where useful

The PSBPL, DMBPL, and ENQPL system initialization parameters can be used in CICS systems that access local DL/I databases.

Limitations

Too low a value for PSBPL or DMBPL may cause excessive deletion and reloading of PSBs and DMBs, and can possibly cause delays in IMS tasks, or even IMS DB abends.

Specification of higher values for the IMS pools requires additional storage if the pools are all used. These situations also increase processor cycle requirements, especially to process database open and close operations, and abends or dumps.

Recommendations

Rather than making IMS DB tasks queue for DL/I threads, consider stopping them earlier by using transaction class controls. In particular, restrict the number of tasks that remain scheduled for relatively long periods, and heavy update tasks.

If you can, define sufficient (virtual) dynamic storage pools to hold all DMBs that may be required, and as many PSBs as you can. The DMBPL should be large enough to hold all DMBs used by the CICS system (including shared databases in the batch region). If the DMBPL system initialization parameter is too small, the

system closes existing unused databases and then opens the required databases. You should expect considerable degradation, therefore, if there is insufficient space for access to several databases. The pool size (specified in 1KB blocks) can be obtained from your IMS DB ACBGEN output.

If more is specified than is needed, the unused DMBPL is available for other uses. If specified too small, a very serious performance problem may result.

The PSBPL minimum size is:

```
[size of the largest PSB] x [DLTHRED (the number of threads)]
```

If this is too small, waits occur during scheduling of the PSB. This is because, when the ceiling is reached, “old” PSBs are removed (on a least-recently-used basis) to make room for the new PSBs, involving additional disk I/O to the ACBLIB.

If there are a large number of PSBs in the system, a higher specification may be desirable if storage constraints allow. This is because, if the PSBPL ceiling is reached, a task requesting a schedule has to wait until a different PSB is free. The old PSB is deleted and the new one loaded at that time.

Take care never to run out of enqueue pool space, because this results in an IMS U0775 pseudoabend. Use of shared databases with batch regions can put particularly high requirements on the enqueue pool space unless the batch program issues checkpoints.

The ENQPL is above the 16MB line and should be specified as very large, because it does not contribute to virtual storage constraints below the line.

How implemented

The PSBPL, DMBPL, and ENQPL parameters are defined in the system initialization table (SIT).

How monitored

The effects of the IMS storage pool settings can be monitored by general IMS DB task throughput and IMS DB I/O rates. Use auxiliary trace to check for queueing for threads, PSBs, and so on.

The maximum usage values are not reported directly by any performance tool, but can be obtained by examining the control fields in the DFHDLP module in a CICS dump. Individual PSB and DMB sizes can be found in ACB generation listings. IMS pseudoabends are also reported in CICS messages to the operating system console and the CSMT destination, and are recorded on the CICS log.

CICS shared database facility

An IMS batch job can access a local DL/I database that is controlled in a CICS region. Any DL/I request from the IMS batch application program is handled through the facilities of CICS instead of IMS DB. This CICS shared database facility is provided by the interregion communication (IRC) component of CICS.

Effects

A shared database region contains a batch application program that processes local DL/I databases, and the application program in the shared database region is scheduled by MVS job management. The job stream for the job specifies the CICS batch region controller. The shared database program uses DL/I CALLs for database references. An application program executing in a shared database region can access only the local DL/I databases that are attached to the CICS online region. Changes made by the program are recorded on the CICS log.

Where useful

The CICS shared database facility can be used with CICS systems that use DL/I databases.

Limitations

This facility can greatly increase contention for a database, particularly if update operations from batch programs are involved. A normal CICS task accesses and enqueues on a small number of records from a database. A batch program may access and enqueue on all the records in the database, effectively locking up the database until the program completes. This can also greatly increase the requirements for storage in the IMS/ESA enqueue pool (see “IMS storage pools (PSBPL, DMBPL, ENQPL)” on page 239).

If batch update operations are required, use of the IMS/ESA or DL/I checkpoint call can free up records when they are updated, but may complicate program restart in the case of a batch program abend. Storage for the dynamic buffer may need to be increased because a large amount of backout information may have to be kept until batch program completion.

Scheduling and turnaround may require the use of CICS shared database, at the expense of real and virtual storage, processor cycles, and additional I/O contention.

Recommendations

In general, CICS using DBCTL performs better than function shipping. Replacing any database owning region (DOR) with a DBCTL owning region improves performance.

Users accessing DL/I databases from CICS via the IMS DBCTL facility should use IMS BMPs rather than CICS shared database.

An accurate SSA= operand in the EXEC DFHDRP step can reduce storage requirements. Use of MVS cross-memory services can minimize SVC processing and reduce MVS common system area (CSA) requirements. If you are using cross-memory services with the shared batch facility, you are recommended to set DFHDRP in the MVS program properties table as non-swappable to prevent transition swaps.

In general, use CICS shared database only when absolutely necessary. Either try to minimize or eliminate update operations and run batch jobs during offpeak times when the system is not busy, or use IMS data sharing.

During this type of operation, report-writing jobs are the most suitable type of batch jobs to execute with CICS online transactions. This is because no batch updates occur and, consequently, there is minimal contention.

If the batch jobs are update jobs, they are most likely to lock out the database from online use until they finish running, which typically takes several minutes.

If it is necessary to run batch update during online operations, do one of the following:

- Run it during low online activity periods
- Close down the online transactions that reference the database
- Inform users of the database that they are most likely to experience an increase in response time during the period of updating from the batch region
- Incorporate frequent checkpoints in batch applications.

You should also review all DL/I PSBs to minimize the contention between batch and online CICS transactions and possibly increase the priority for online transactions versus the partition control task.

How implemented

The TYPE=IRCBCH macro in the terminal control table and the IRCSTRT system initialization parameter determine whether the shared database facility is enabled at CICS initialization. The facility can be turned on or off with CEMT INQ or SET IRC commands.

How monitored

The CICS DL/I statistics show database calls and program usage.

CICS attachment facility

The CICS attachment facility provides a multi-thread connection to DB2. The connections between CICS and DB2 are called threads. There are three types of thread:

Command threads.
One or more threads can be reserved for command usage. It is used for DB2 commands only; for example, the DSNC-DIS command. During periods of heavy command usage, requests for command threads may be transferred to pool threads.

Pool threads
Pool threads are normally used for low volume transactions, and transactions that overflow from either entry threads or command threads. These threads are created when needed and terminate immediately when unused.

Entry threads
These threads are intended for high-volume, high-priority transactions. Each thread is associated with a particular application plan, and the threads are reusable.

Entry threads can be defined as **protected** which means that they are not terminated immediately if unused. They are terminated after two consecutive 30-second periods of inactivity. Many CICS transactions can use the same protected threads and avoid the overhead involved in creating and terminating the thread for each transaction.

The resource control table (RCT) of the CICS attachment facility defines the authorization and access attributes on a transaction and transaction group basis.

Effects

The TWAIT parameter of the RCT defines whether the requests for a thread should be queued, abended, or sent to the pool thread in the case of a shortage of entry or command threads. If TWAIT=YES is specified instead of TWAIT=POOL, the transaction is queued rather than sent to the pool thread. Using TWAIT=YES, therefore, avoids the thread initialization and termination overhead. If a transaction is made to wait because of the lack of entry threads, a queueing arrangement is necessary. This is done by the CICS attachment facility. The advantages of this are that, once the entry thread finishes its current piece of work, it continues with the next transaction immediately.

| You can optimize performance between CICS and DB2 by adjusting the transaction
| class limits, MXT system parameters of CICS and the TWAIT, THRDMAX, THRDM,
| THRDA, and DPMODE parameters of the RCT.

Where useful

In a high-volume, highly-utilized system.

How implemented

TWAIT is defined in the resource control table (RCT) of the CICS attachment facility.

MAXACTIVE is an attribute of the transaction class definition. The maximum number of transaction class can be adjusted dynamically by CEMT or EXEC CICS SET TRANCLASS MAXIMUM.

How monitored

The following facilities are available to monitor CICS attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with entries in the resource control table (RCT).
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

CICS attachment facility (THRDMAX, THRDM and THRDA)

THRDMAX, THRDM and THRDA are parameters the resource control table. They can be set for each of the three types of thread (see page 243 for more information). THRDMAX specifies the maximum number of threads between CICS and DB2. THRDM specifies the maximum value that can be specified for THRDA which, in turn, specifies the maximum number of active DB2 threads. THRDA is modified dynamically.

The sum of all the active threads from TSO users, all CICS and IMS systems and other systems accessing DB2 should not exceed NUMCONCR. Otherwise, the result could be unpredictable response times. When this occurs, a CICS attachment facility “create thread” request is queued by DB2, and the CICS transaction is placed is a wait state until a thread is available.

Note: NUMCONCR is a DB2 parameter, specified in ZPARMS, and it defines the number of concurrent threads for all of DB2.

Effect

Each thread linking CICS to DB2 has a corresponding TCB in the CICS address space. Too many TCBs per address space involve the MVS dispatcher scanning the TCBs to identify an active TCB. If there is a large number of TCBs then there may be a significant cost of processor time.

When the total number of active threads in the system reaches THRDMAX – 2, the attach scans its TCBs and detaches all of the TCBs that are inactive. It therefore needs to reattach TCBs as needed. A large number of TCB attaches and detaches have a major performance implication.

Where useful

If there are a large number of threads per CICS region then it is worth reducing the number of threads per CICS region.

Ensuring that the sum of all the THRDAs stays below the THRDMAX – 2 value prevents this task purge from happening.

Limitations

Increasing the THRDMAX value and setting up an additional CICS system with access to the same DB2 system may require increasing the NUMCONCR parameter of DB2.

Recommendations

If the number of threads in a given CICS region exceeds 20, it is worth considering the redistribution of threads.

The user should check that the sum of the THRDA from all the command, pool and entry threads is less than or equal to the THRDMAX – 2.

How implemented

For pool thread environments, this can be achieved simply by reducing the value of both THRDM and THRDA.

For a protected entry thread environment, implementation involves reviewing the number of application plans and, if possible, reducing the number of plans by combining infrequently used ones while balancing the issues of plan size and security.

Initially, you should start with one thread per plan. In a high-volume transaction processing environment, you can estimate the initial number by using the occupancy time of a thread by a transaction and multiplying it with the expected transaction rate. For example, an occupancy time of 0.2 seconds and a transaction rate of 20 transactions per second (0.2 x 20) would give an initial thread number of between three and four.

Finally, if there are a large number of threads per CICS region, then it is advisable to reduce the number of threads available per CICS region. The optimum number of threads is unlikely to exceed 15 to 20 per CICS region.

How monitored

The following facilities are available to monitor CICS attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with entries in the resource control table (RCT).
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

CICS attachment facility (DPMODE)

DPMODE is a parameter of resource control table (RCT) of the CICS attachment facility that can be specified for both the pool and entry threads. This parameter controls the priority of the CICS TCBs. There are three options: DPMODE=HIGH, DPMODE=LOW, and DPMODE=EQUAL. (See the *DB2 Administration Guide* for more information.)

Effects

When DPMODE is specified as HIGH, transactions run at a higher priority than CICS thus saving virtual storage, releasing locks, and avoiding other transactions deadlocking or timing out. However, if all threads are specified with DPMODE HIGH, CICS itself may be effectively at too low a priority.

Where useful

Setting DPMODE=HIGH is useful for high-priority and high-volume transactions.

Limitations

A complex SQL call could spend a long time in DB2, and the CICS TCB may not be dispatched.

Recommendations

Set DPMODE=HIGH for your transactions with the highest weighted average number of SQL calls. The highest weighted average is equal to the number of SQL calls per transaction multiplied by the frequency of transaction. Set DPMODE=LOW or EQUAL for other transactions. If the CPU usage per call is high, you should not set DPMODE=HIGH.

How implemented

DPMODE is a parameter of the RCT of CICS attachment facility.

How monitored

The following facilities are available to monitor CICS attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with entries in the resource control table (RCT).
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

Chapter 20. Journaling

This chapter includes the following topics:

Activity keypoint frequency (AKPFREQ)

The activity keypoint frequency value (AKPFREQ) specifies the number of physical outputs to the CICS system log before an activity keypoint is to be taken. A keypoint is a snapshot of in-flight tasks and tables (such as the destination control table (DCT), and temporary storage table (TST)) in the system at that time. During emergency restart, CICS only needs to read back for records for those tasks identified in a keypoint. CICS reads the system log backward until the first activity keypoint is encountered. This activity keypoint is analyzed to determine if the system log should be read backward further to gather data for in-flight tasks.

In summary, the AKPFREQ value determines the amount of writing on the journal between keypoint frequencies and, therefore, the amount that may need to be read back on an emergency restart. It also determines the extent to which the journal data is interspersed with additional data for keypoints.

Limitations

Increasing the AKPFREQ value has the following effects:

- Restart and XRF takeover times tend to increase.
- The size requirements of the restart data set (DFHRSD) may be increased.
- The amount of data that is written to the log is reduced.

Decreasing the AKPFREQ value has the following effects:

- Restart time may be reduced. This is particularly important in high-availability systems such as those running with XRF=YES (see “Using extended recovery facility (XRF)” on page 311).
- Task wait time and processor cycles tend to increase.
- Paging may increase.
- More data is written to the log.

Taking a keypoint imposes an overhead on the running system. Paging increases at keypoint time because many control blocks are scanned.

Setting the frequency to zero makes emergency restart impossible.

Recommendations

If you set AKPFREQ too high and thus make your keypoint frequency too low, the writing of the keypoints causes the system to slow down for a short time. If you set AKPFREQ too low and make your keypoint frequency too high, you may get a short emergency restart time but you also increase the amount of data on the log because a higher proportion of it is keypoint data.

Having a low AKPFREQ value does not help in reducing the log read time if the system contains long-running tasks which do not take regular syncpoints when they update recoverable resources. If the system contains this sort of task, it has to read back the LOG all the way to the last syncpoint (or transaction start) for that task.

Set AKPFREQ to a value that allows activity keypoints no more often than once every 15 minutes. If your system takes keypoints more frequently, increase AKPFREQ.

How implemented

Activity keypoint frequency is determined by the AKPFREQ system initialization parameter. The CSKP transaction and the DFHAKP program must also be installed (IBM-supplied group DFHAKP). AKPFREQ can be altered with the CEMT SET SYSTEM[AKP(value)] while CICS is running.

How monitored

A message, DFHJC5801, is written to the CSMT transient data destination each time a keypoint is taken. Use CICS transaction statistics (see page 425) to see the frequency of execution of the CSKP transaction.

CICS journaling (BUFSIZE, SYSWAIT=STARTIO|ASIS)

The buffer size (BUFSIZE) operand in the journal control table indicates the size of each buffer used in journal output. The size for the buffers ranges from 256 through 32760 bytes. Both buffers of each journal are of equal size and are created and acquired at CICS initialization.

BUFSIZE limits the maximum size of a journal block. Thus, it also implies a limit on the maximum size of any single record written to that journal. Journal control builds a label record at the front of the buffer and the remaining space in the buffer constitutes the maximum size of any single record.

Note: If no change is made to the value you specified for BUFSIZE in releases before CICS/ESA 4.1, a GETMAIN will result in twice the former amount of storage being acquired.

Effects

Journal records are blocked, variable-length records. They are passed to the buffers in the following way:

1. Records are added to the first (current) buffer until such time that output is required. This could be because:
   - An immediate (STARTIO) write has been requested for a record.
   - A synchronous (WAIT) request was received and one second has elapsed.

- The current buffer is full.

2. The two buffers are swapped, the current buffer becoming the alternate, and the alternate becoming the current.
3. Output is performed on the alternate buffer, containing the records just added.
4. While output is in progress, new records may also be added to the current buffer.
5. If the current buffer becomes full before output has completed on the alternate buffer, the 'buffer full' statistic is incremented by one, and any task attempting to put a record into that journal is held until the output is completed.

If SYSWAIT=STARTIO (the default) has been coded in the JCT, all synchronizing requests from CICS management modules will force immediate output. In general, this will only affect the system log, as most requests from CICS management modules are directed to the system log. However, CICS management modules also write to user journals when they have been specified as forward recovery logs.

Normally, SYSWAIT=STARTIO should be chosen to minimize transaction response times. However, if the frequency of output to the journal becomes so high that the device becomes overloaded, code SYSWAIT=ASIS in the JCT. Output is then performed as necessary according to the criteria in the the list in point 1 on page 250 above.

If SYSWAIT=ASIS is preferred, it is recommended that you code it only for user journals. Use of this option may reduce any contention on a busy device and provide very small CPU savings. Response times for transactions using journals with this option are likely to be longer than when using the default.

Limitations

BUFSIZE must not exceed the physical capacity of the journal device that is going to receive the data. There is a maximum limit of 32760 bytes and, on DASD devices, the limit may be smaller because of a smaller track capacity.

On the other hand, BUFSIZE must be at least large enough to hold the maximum sized single journal record (together with the label) that may be written to the journal.

Recommendations

In most systems it is preferable to use the default (SYSWAIT=STARTIO) thus minimizing response time. For the system log, use a BUFSIZE such that 'buffer full' does not occur. If possible, all user journals should also have a BUFSIZE such that 'buffer full' does not occur.

The number of 'buffer full', as reported in CICS statistics, should be as near to zero as possible for the CICS system log. The BUFSIZE should be made large enough to ensure this. The virtual storage cost of the buffer is likely to be much less than the virtual storage cost of increased transaction response times incurred when ‘buffer full’ occurs.

For programming information on the format of journal records, see the *CICS/ESA Customization Guide*.

A busy CICS journal, particularly the system log, has typically a very high device utilization rate recorded by RMF, and should, therefore, be the only high-activity data set on a DASD volume.

How implemented

Journal output characteristics are defined with the BUFSIZE, and SYSWAIT=STARTIO|ASIS options in the DFHJCT TYPE=ENTRY macro.

How monitored

The CICS journal control statistics show activity and blocking of records to the defined journals. The following information is given in these statistics (see page 372):

- The number of output requests made
- The number of blocks written
- The average length of blocks written
- The number of times the buffer was full (that is, the current buffer had filled before output has completed on the alternate buffer).

Small journal size and long running transactions that update protected resources and do not syncpoint, may cause the start of a long running transaction to become 'lost' from the currently accessible system log. This makes transaction backout impossible.

Journal volume switches (JOUROPT=AUTOARCH|PAUSE)

To allow you to continue writing to journals while archiving is in progress, two disk data sets are required for every journal. CICS archives the filled data set while writing to the second data set.

When a journal data set is filled, it is closed and CICS automatically archives the data set to either tape or disk. The journal data set is not reused until archiving is complete.

The only need for operator intervention should be to mount a new tape.

Effects

If the journal was specified with the PAUSE option in its journal control table (JCT) entry, at the switch to the second data set, CICS sends either message DFHJC4583 or DFHJC4586. This message indicates that the specified journal disk is about to be overwritten by output. The operator has to respond to this message before the next switch can take place. If the operator has not responded before the second extent becomes full, the JCT PAUSE option causes logging to cease until the operator has responded.

Recommendations

If AUTOARCH is used, two journal data sets are required.

Automatic journal archiving (JOUROPT=AUTOARCH in the JCT) minimizes operator intervention and the possibility of the second data set filling before the first has been archived.

Forcing a journal volume switch during non-peak periods can minimize the effect of the wait. Use two tape drives, with maximum tape reel data capacity, to minimize the effect with tape journaling. Using two tape drives eliminates the rewind and remount time.

How implemented

A switch can be forced on a data set by using EXEC CICS SET JOURNALNUM ADVANCE, or with CEMT SET JOURNALNUM ADVANCE. See the *CICS/ESA Recovery and Restart Guide* and the *CICS/ESA Resource Definition Guide* for further information on the JCT specifications, and the *CICS/ESA CICS-Supplied Transactions* manual for further information on the CEMT transaction.

How monitored

Journal archiving information is part of journal statistics (see page 372).

Chapter 21. Virtual and real storage

This chapter includes the following topics:

This chapter discusses performance tuning issues related to virtual and real storage.

CICS storage management works in units of pages that are normally identical to the operating system page size. Think about choosing area sizes so that they fit properly onto pages.

Allocations that specify a number of storage areas should be set to fill all pages used. Storage size allocations should be defined at a size that optimizes page usage. Allocation of storage areas that are exactly divisible by the page size completely fill the pages.

Tuning CICS virtual storage

The CICS virtual storage tuning process consists of several steps that you should take in the following order:

1. Understand the contents of the CICS address space. To determine what is good or bad in your system, you must first fully understand the contents of the CICS address space and what components of the system affect the size of each of the areas. See Appendix C, “MVS and CICS virtual storage” on page 493 for a description of the CICS address space.
2. Measure the CICS address space using one of the following tools to determine the approximate sizes of each of the areas:
   - CICS formatted dump (loader domain and storage domain only) to look at the CICS region
   - CICS storage statistics
   - The sample statistics program (DFH0STAT) to provide selected statistical information for estimating the size of your DSAs.

3. Using the results of the above measurements, determine if each of the areas of the CICS address space is within the expected sizes and select the areas that seem to be the most out of line from your expectations. Concentrate on these items; do not waste time on areas that represent only a small amount of storage improvement.
4. Evaluate the guidance given in the following chapter to see whether it is applicable to your installation.

Splitting online systems: virtual storage

A method of increasing the virtual storage available to a CICS system is to split the system into two or more separate address spaces. Splitting a system can also allow you to use multiprocessor complexes to the best advantage because a system can then operate on each processor concurrently. Splitting systems can also provide higher availability; see "Splitting online systems: availability" on page 181. See page 279 for information on using intercommunication facilities.

If data, programs, or terminals must be shared between the systems, CICS provides intercommunication facilities for this sharing. Two types of intercommunication are possible:

1. **Intersystem communication (ISC)**. ISC is implemented through the VTAM LU6.1 or LU6.2. These give program-to-program communication with System Network Architecture (SNA) protocols. ISC includes facilities for function shipping, distributed transaction processing, and transaction routing.
2. **Multiregion operation (MRO)**. MRO is implemented through MVS cross-memory facilities. An alternative method is to use operating system supervisor calls (SVCs). For communication across MVS images within a SYSPLEX, MRO/XCF is implemented using the MVS cross-system coupling facility. It includes function shipping, distributed transaction processing, and transaction routing.

The definition of too many MRO sessions can unduly increase the processor time used to test their associated ECBs. Use the CICS-produced statistics (see "ISC/IRC system and mode entries" on page 361) to determine the number of MRO sessions defined and used. For more detailed information on ISC and MRO, see the *CICS/ESA Intercommunication Guide*.

MRO also allows you to use multiprocessors more fully, and the multiple address spaces can be dispatched concurrently. MRO is implemented primarily through changes to CICS resource definitions and job control statements for the various regions. To relieve constraints on virtual storage, it may be effective to split the CICS address space in this manner.

Function shipping allows you to define data sets, transient data, temporary storage, IMS databases, or interval control functions as being remote. This facility allows applications to request data set services from a remote region (that is, the other CICS address space where the data sets are physically defined). Heavy use of VSAM and DL/I resources requires large amounts of virtual storage. If, for example, 500 VSAM KSDS data sets are removed to a remote region from the region where the application is being run, this can potentially save more than one megabyte.

The DL/I call and EXEC interfaces are supported for function shipping. CICS handles the access to remote resources and returns the requested items to a program without the need for recoding the program. Use of DL/I through DBCTL is usually a better alternative, and IMS data sharing might also be considered.

Distributed transaction processing allows direct communication between one application program and another application program, on a "send/receive" basis, much as a program communicates with a terminal. See the *CICS/ESA Distributed Transaction Programming Guide* for information about DTP.

Transaction routing allows a terminal owned by one CICS region to run a transaction that resides in another region, as if that transaction resided in the terminal-owning region.

Where useful

Most CICS systems can be split.

Limitations

Splitting a CICS region requires increased real storage, increased processor cycles, and extensive planning.

If you only want transaction routing with MRO, the processor overhead is relatively small. The figure is release- and system-dependent (for example, it depends on whether you are using cross-memory hardware), but for safety, assume a total cost somewhere in the range of 15–30KB instructions per message-pair. This is a small proportion of most transactions: commonly 10% or less.

The cost of MRO function shipping can be very much greater, because there are normally many more inter-CICS flows per transaction. It depends greatly on the disposition of resources across the separate CICS systems.

MRO can affect response time as well as processor time. There are delays in getting requests from one CICS to the next. These arise because CICS terminal control in either CICS system has to detect any request sent from the other, and then has to process it; and also because, if you have a uniprocessor, MVS has to arrange dispatching of two CICS systems and that must imply extra WAIT/DISPATCH overheads and delays.

The system initialization parameter ICVTSD (see page 204) can influence the frequency with which the terminal control program is dispatched. Another system initialization parameter is MROLRM, which should be coded yes if you want to establish a long-running mirror task. This saves re-establishing communications with the mirror transaction if the application makes many function shipping requests in a unit of work. A value in the range 300–1000 milliseconds is typical in non-MRO systems, and a value in the range 150–300 is typical for MRO systems (and even lower if you are using function-shipping).

You also have to ensure that you have enough MRO sessions defined between the CICS systems to take your expected traffic load. They do not cost much in storage and you certainly do not want to queue. Examine the ISC/IRC statistics to ensure that no allocates have been queued, also ensure that all sessions are being used.

Other parameters, such as MXT, may need to be adjusted when CICS systems are split. In an MRO system with function shipping, tasks of longer duration might also require further adjustment of MXT together with other parameters (for example, file string numbers, virtual storage allocation). Finally, if you plan to use MRO, you may want to consider whether it would be advantageous to share CICS code or application code using the MVS link pack area (LPA). Note that this is to save real storage, not virtual storage, and other non-CICS address spaces. Use of LPA for the eligible modules in CICS is controlled by the system initialization parameter, LPA=YES; this tells CICS to search for the modules in the LPA. For further information on use of LPA, see “Using modules in the link pack area (LPA/ELPA)” on page 269.

Recommendations

To tune CICS to get more virtual storage, you must first tune MVS and then CICS. If, after you have tuned MVS common virtual storage, you still cannot execute CICS in a single address space, you must then consider splitting the CICS workload into multiple address spaces. Many installations find it convenient to split their CICS workload into multiple independent address spaces, where their workload is easily definable and no resource sharing is required. If it is easy to isolate application subsystems and their associated terminals, programs, and data sets, it is reasonable to split a single CICS address space into two or more independent address spaces. They become autonomous regions with no interactions.

A system can be split by application function, by CICS function (such as a data set owning or terminal owning CICS), or by a combination of the two functions. Ideally, you should split the system completely, with no communication required between the two parts. This can reduce overheads and planning. If this is not possible, you must use one of the intercommunication facilities.

You can provide transaction routing between multiple copies of CICS. If additional virtual storage is needed, it would be reasonable, for example, to split the AOR into two or more additional CICS copies. When you have split the system either partially or completely, you can reduce the amount of virtual storage needed for each region by removing any unused resident programs. One consequence of this is reduce the size of the relevant DSA.

Admittedly, MRO uses additional processor cycles and requires more real storage for the new address spaces. Many installations have several megabytes of program storage, however, so the potential virtual storage savings are significant.

You should also remember that only a local or remote PSB can be scheduled at one time with function shipping, affecting the integrity of the combined databases. Distributed transaction processing can allow for transactions in both systems to concurrently schedule the PSBs.

MRO generally involves less overhead because the processing of the telecommunications access method is avoided. VTAM logons and logoffs can provide an alternative to transaction routing if the logons and logoffs are infrequent.

How implemented

You must define resources in the CSD (CICS system definition) data set, such as program files and terminal definitions. You must also check links to other systems, together with the connection and session definitions that substantiate such links.

MRO across the MVS sysplex

The CICS IRC facility that supports MRO is enhanced to exploit the XCF of MVS/ESA, to provide dynamic add of connections, and to rationalize MRO security.

The main benefit of adding XCF/MRO to the CICS interregion communication facility is to provide efficient and flexible CICS-to-CICS communications in an MVS sysplex environment. By exploiting the MVS cross-system coupling facility, CICS supports MRO links between MVS images, enabling you to use transaction routing, function shipping, and distributed program link across MRO links in a sysplex environment, replacing the need to use CICS ISC links through VTAM for these functions. XCF/MRO consumes much less CPU resources than ISC. A sysplex consists of multiple MVS systems, coupled together by hardware elements and software services. In a sysplex, MVS provides a platform of basic multisystem services that multisystem applications like CICS can exploit. As an installation's workload grows, additional MVS systems can be added to the sysplex to enable the installation to meet the needs of the increased workload.

You can also use XCF/MRO for distributed transaction processing, provided the LU6.1 protocol is adequate for your purpose.

Maximum task specification (MXT)

The MXT system initialization parameter limits the total number of concurrent user tasks in the CICS system. It also affects the amount of storage allocated to the kernel stack segment.

Effects

MXT primarily controls virtual storage usage, particularly to avoid short-on-storage (SOS) conditions. It also controls contention for resources, the length of queues (this can avoid excessive processor usage), and real storage usage.

MXT controls the number of user tasks that are eligible for dispatch. When MXT is set (either at startup, when an EXEC CICS SET SYSTEM command is processed, or when using a CEMT transaction) the kernel and dispatcher attempt to preallocate sufficient control blocks to guarantee that MXT user tasks can be created concurrently. The majority of the storage used in this preallocation is obtained from the CDSA or ECDSA, although a small amount of MVS storage is required for each task (approximately 256 bytes above the 16MB line, and 32 bytes below the 16MB line for each user task). It is interrelated with the DSA size limits that you set (DSALIM, EDSALIM).

Limitations

If you set MXT too low, throughput and response time can suffer when system resources (processor, real storage, and virtual storage) are not constrained.

If you set MXT too high at startup, CICS forces a smaller maximum number of tasks consistent with available storage.

If you set MXT too high while running, you get the error message: “CEILING REACHED.”

For MRO considerations, read about the secondary effects of the region exit interval (ICV) on page “Region exit interval (ICV)” on page 187.

Recommendations

Initially, set MXT to the number of user tasks you require concurrently in your system by totaling the following:

- The number of concurrent long-running tasks
- Each terminal running conversational tasks
- An estimate of the number of concurrent tasks from terminals running nonconversational tasks
- An estimate of the number of concurrent nonterminal tasks.

How implemented

The MXT system initialization parameter has a default value of 5, and a minimum setting of 1. It can be altered with either CEMT or EXEC CICS SET SYSTEM MAXTASKS commands while CICS is running.

How monitored

The CICS transaction manager statistics show the number of times the MXT ceiling has been reached.

Transaction class (MAXACTIVE)

Transaction classes give you a mechanism to limit the number of CICS tasks within your system. By spreading your tasks across a number of transaction classes and controlling the maximum number of tasks that can be dispatched within each transaction class, you can control resource contention between tasks and limit the number of tasks that CICS considers eligible for dispatching at task attach.

Effects

Together with MXT, transaction classes control the transaction “mix”, that is, it ensures that one type of transaction does not monopolize CICS.

When the number of tasks within a class is at the specified ceiling, no additional tasks within that class are attached until one of them terminates.

Transaction classes can be used to force single-threading of a few tasks, either to avoid ENQ interlocks or because of the excessive effect of several such tasks on the rest of the system.

Limitations

Transaction classes are unsuitable in normal use for conversational transactions, because the (n+1) user may be locked out for a long time.

You should note that when a very large number of tasks are queued, some additional CPU is consumed during the queueing of a task.

If TRANCLASS is specified with transaction CATD, the MAXACTIVE attribute of the transaction class must have a value of at least two in the corresponding field to prevent all the CATD transactions stacking up behind the one in the ECB wait during an emergency restart. See the *CICS/ESA Resource Definition Guide* for more details of TRANCLASS.

Recommendations

The MAXACTIVE attribute of the transaction class definition can be used to control a specific set of tasks that may be heavy resource users, tasks of lesser importance (for example, “Good morning” broadcast messages), and so on, allowing processor time or storage for other tasks.

By selecting transaction classes and their MAXACTIVE values, you can control the mix of transactions; that is, you can ensure that one type of transaction does not monopolize CICS. In particular, you can restrict the number of “heavyweight” tasks, the load on particular data sets or disk volumes, and the printer load on lines. For example, you can use transaction classes to isolate “difficult” tasks, or put all user tasks into separate classes. Suggested classes are simple enquiries, complex enquiries or short browses, long browses, short updates, long updates. Separate nonconversational tasks from conversational tasks. If you need to single-thread non-reentrant code, use ENQ for preference.

Using transaction classes can be useful for particularly high-resource-consuming tasks that do not exceed MAXACTIVE ceiling frequently, but should not be implemented for normal tasks or for design reasons such as serializing a function within a particular task. Application design should be reviewed as an alternative in these cases.

How implemented

You specify the maximum number of tasks in each transaction class using the MAXACTIVE attribute. You specify the value of the class associated with a particular task using the CEDA transaction definition with the TRANCLASS keyword. Most CICS Cxxx transaction identifiers are not eligible.

MAXACTIVE values can be changed using the CEMT SET TRANCLASS(classname) MAXACTIVE(value) or EXEC CICS SET TRANCLASS() MAXACTIVE() commands.

How monitored

If you have divided your tasks into classes, you can use the CEMT INQUIRE TCLASS command to provide an online report. The CICS transaction class statistics show the number of times that the number of active transactions in the transaction class reached the MAXACTIVE value (“Times MaxAct”).

Transaction class purge threshold (PURGETHRESH)

The PURGETHRESH attribute of the transaction class definition limits the number of tasks which are newly created, but cannot be started because the MAXACTIVE limit has been reached for the associated transaction class. These tasks are queued by the transaction manager domain in priority order until they obtain class membership.

They occupy small amounts of storage, but if the queue becomes very long CICS can become short-on-storage and take a considerable time to recover. Systems where a heavy transaction load is controlled by the TRANCLASS mechanism are most prone to being overwhelmed by the queue.

The tasks on the queue are not counted by the MXT mechanism. MXT limits the total number of tasks that have already been admitted to the system within TRANCLASS constraints.

Effects

The length of the queue of tasks waiting to be started in a TRANCLASS is limited by the PURGETHRESH attribute of that class. Any new transaction which would cause the limit to be reached is abended with the abend code AKCC. Tasks that were queued before the limit was reached are allowed to continue waiting until they can be executed.

Where useful

The PURGETHRESH attribute should be specified only where the transaction load in a TRANCLASS is heavy. This is the case in a system which uses a terminal-owning region (TOR) and multiple application-owning regions (AORs) and where the TRANCLASSes are associated with the AORs and are used to control the numbers of transactions attempting to use the respective AORs. In this configuration, an AOR can slow down or stall and the associated TRANCLASS fills (up to the value defined by MAXACTIVE) with tasks that are unable to complete their work in the AOR. New transaction are then queued and the queue can grow to occupy all the available storage in the CICS DSA within a few minutes, depending on the transaction volume.

Recommendations

The size of each entry in the queue is the size of a transaction (256 bytes) plus the size of the TIOA holding any terminal input to the transaction. There can be any number of queues, one for each TRANCLASS that is installed in the TOR.

You can estimate a reasonable size for the queue by multiplying the maximum length of time you are prepared for users to wait before a transaction is started by the maximum arrival rate of transactions in the TRANCLASS.

Make sure that the queues cannot occupy excessive amounts of storage at their maximum lengths.

The queuing limit should not be set so low that CICS abends transactions unnecessarily, for example when an AOR slows down due to a variation in the load on the CPU.

How implemented

The PURGETHRESH attribute of a TRANCLASS is used to set the limit of the queue for that transaction classes. The default action is not to limit the length of the queue.

Note that the CEMT SET TRANCLASS(name) PURGETHRESH(p) command can be used to change the purge threshold of a transaction class online.

How monitored

To monitor the lengths of the queues for each transaction class you should use CICS transaction class statistics. Many statistics are kept for each transaction class. Those that are particularly relevant here are:

XMCPI
Number of transactions abended AKCC because the size of the queue reached the PURGETHRESH limit.

XMCPQT
The peak number of transactions in the queue.

XMCTAPT
The number of times the size of the queue reached the PURGETHRESH limit.

You can also tell how many tasks are queued and active in a transaction class at any one time by using the CEMT INQUIRE TRANCLASS command.

You can monitor the number of AKCC abends in the CSMT log. These abends indicate the periods when the queue limit was reached. You must correlate the transaction codes in the abend messages with the transaction classes to determine which limit was being reached. The tasks on the queue are not counted by the MXT mechanism. MXT limits the total number of tasks that have already been admitted to the system within TRANCLASS constraints.

Task prioritization

Prioritization is a method of giving specific tasks preference in being dispatched.

Priority is specified by terminal in a CEDA TERMINAL definition (TERMPRIORITY) or the terminal control table (TCT), by transaction in a CEDA TRANSACTION definition (PRIORITY) and by operator in the signon table (SNT) (OPPRTY).

The overall priority is determined by summing the priorities in all three definitions for any given task, with the maximum priority being 255.

```
TERMPRIORITY+PRIORITY+OPPRTY => 255
```

The value of the system initialization parameter, PRTYAGE also influences the dispatching order, for example, PRTYAGE=1000 causes the task's priority to increase by 1 every 1000ms it spends on the ready queue.

Effects

With CICS/ESA the dispatching priority of a task is reassessed each time it becomes ready for dispatch, based on clock time as well as defined priority.

A task of priority n+1 that has just become ready for dispatch is usually dispatched ahead of a task of priority n, but only if PRTYAGE milliseconds have not elapsed since the latter last became ready for dispatch.

Thus, a low priority task may be overtaken by many higher priority tasks in a busy system, but eventually arrives at the top of the ready queue for a single dispatch.

The lower the value of PRTYAGE, the sooner this occurs.

Where useful

Prioritization is useful for browsing tasks, and tasks that use a lot of processor time. Input/Output bound tasks can take the required amount of CPU, and move on to the next read/write wait. CPU-intensive tasks take higher priority over the less intensive tasks.

Prioritization can be implemented in all CICS systems. It is more important in a high-activity system than in a low-activity system. With careful priority selection, an improvement in overall throughput and response time may be possible.

Input/Output bound tasks can take the required amount of CPU, and move on to the next read/write wait. CPU-intensive tasks take higher priority over the less intensive tasks.

Prioritization can minimize resource usage of certain resource-bound transactions.

Limitations

Prioritization increases the response time for lower-priority tasks, and can distort the regulating effects of MXT and the MAXACTIVE attribute of the transaction class definition.

Priorities do not affect the order of servicing terminal input messages and, therefore, the time they wait to be attached to the transaction manager.

Because prioritization is determined in three sets of definitions (terminal, transaction, and operator), it can be a time-consuming process for you to track many transactions within a system.

CICS prioritization is not interrupt-driven as is the case with operating system prioritization, but simply determines the position on a ready queue. This means that, after a task is given control of the processor, the task does not relinquish that control until it issues a CICS command that calls the CICS dispatcher. After the dispatch of a processor-bound task, CICS can be tied up for long periods if CICS requests are infrequent. For that reason, prioritization should be implemented only if MXT and the MAXACTIVE attribute of the transaction class definition adjustments have proved to be insufficient.

Recommendations

Use prioritization sparingly, if at all, and only after you have already adjusted task levels using MXT and the MAXACTIVE attribute of the transaction class definition.

It is probably best to set all tasks to the same priority, and then prioritize some transactions either higher or lower on an exception basis, and according to the specific constraints within a system.

Do not prioritize against slow tasks unless you can accept the longer task life and greater dispatch overhead; these tasks are slow, in any case, and give up control each time they have to wait for I/O.

Use small priority values and differences. Concentrate on transaction priority. Give priority to control operator tasks rather than the person, or at least to the control operator's signon ID rather than to a specific physical terminal (the control operator may move around).

Consider for **high** priority a task that uses large resources. However, the effects of this on the overall system need careful monitoring to ensure that loading a large transaction of this type does not lock out other transactions.

Also consider for **high** priority those transactions that cause enqueues to system resources, thus locking out other transactions. As a result, these can process quickly and then release resources. Examples of these are:

- Using intrapartition transient data with logical recovery
- Updating frequently used records
- Automatic logging
- Tasks needing fast application response time, for example, data entry.

Lower priority should be considered for tasks that:

- Have long browsing activity
- Are process-intensive with minimal I/O activity
- Do not require terminal interaction, for example:
  - Auto-initiate tasks (except when you are using transient data intrapartition destinations or queues with a destination facility of "terminal" (DESTFAC=TERMINAL) in the destination control table (DCT), and the TRIGGER level is greater than zero)
  - Batch update controlling tasks.

PRTYAGE should usually be left to its default value, unless certain transactions get stuck behind higher priority transactions during very busy periods.

How implemented

You specify the priority of a **transaction** in the CEDA TRANSACTION definition with the PRIORITY keyword. You specify the priority for a **terminal** in the CEDA terminal definition with the TERMPRIORITY keyword or in the TCT with the TRMPRTY operand. You specify the priority for an **operator** in the SNT with the OPPRTY operand.

PRTYAGE is a system initialization parameter.

How monitored

There is no direct measurement of transaction priority. Indirect measurement can be made from:

- Task priorities
- Observed transaction responses
- Overall processor, storage, and data set I/O usage.

Simplifying the definition of CICS dynamic storage areas

CICS allocates dynamic storage areas automatically. This removes the need to specify the size of each individual dynamic storage area. You need specify only the overall limits within which CICS can allocate storage for these areas.

CICS uses eight separate dynamic storage areas:

CDSA
RDSA
SDSA
UDSA

ECDSA
ERDSA
ESDSA
EUDSA

To facilitate continuous operations, and to simplify CICS/ESA system management, the individual DSA sizes are determined by CICS, and can be varied dynamically by CICS as the need arises. You simply specify how much storage that CICS is to use for the DSAs in two amounts—one for the four DSAs above the 16MB boundary, and the other for the four DSAs below. Automatic sizing within the specified limits removes the need for restarts to change DSA sizes. You can also vary the overall limits dynamically, using either the CEMT master terminal command, or an EXEC CICS SET command. See “The dynamic storage areas” on page 503 for more information about considerations regarding the size you should set for the DSALIM and EDSALIM parameters.

Extended Dynamic Storage Areas

Conceptually, you should view the system initialization parameter, EDSALIMIT, as limiting the size of one large storage pool where each of the DSAs above the line (ECDSA, ESDSA, EUDSA, ERDSA) acquire space. The unit of allocation is 1MB extents. An allocated extent can be used by only the owning EDSA (EDSAs cannot share a given extent). If there is not enough space within the allocated extents to satisfy a request, additional extents are acquired as necessary unless the EDSALIMIT has been reached.

In situations where one of the EDSAs attempts to acquire an additional extent and there are no free extents, empty extents belonging to other EDSAs are used. Program compression may be triggered when the EDSALIMIT is approached and there are few free or empty extents available. The EUDSA no longer contains programs, and so program compression does not occur in it. The other EDSAs are evaluated individually to determine if program compression is required.

Estimating the EDSALIMIT

Specify the EDSALIMIT so that there is sufficient space to accommodate all the EDSAs.

- The EDSAs (ECDSA, ESDSA, EUDSA and ERDSA) are managed by CICS as part of the EDSALIMIT. Because the EDSAs are managed in 1 megabyte increments (extents), it is important to allow for fragmentation and partially used extents by rounding up the value of the EDSALIMIT accordingly. Because there are 4 extended DSAs, consider rounding up each EDSA's requirement to a megabyte boundary.
- If TRANISO=NO, you must allow 64K per concurrent active task for the EUDSA. The safest estimate is to assume MXT as the number of concurrent active tasks. If your applications use more than 64K per task, you must adjust the formulas accordingly (use multiples of 64K increments if adjusting the formula).
- If TRANISO=YES, you must allow 1 megabyte per concurrent active task for the EUDSA. Again, the safest estimate would be to assume MXT as the number of concurrent active tasks. If your applications use more than 1 meg per task, you must adjust the formulas accordingly (use multiples of 1 meg increments if adjusting the formula).

If you are migrating from a CICS/ESA 3.3 system, two methods of estimating the EDSALIMIT are shown below. Information can be obtained by looking at your current storage manager statistics (see the DSA limit in the storage manager statistics, dynamic storage areas and in the task subpools).

Kernel stack storage is allocated out of EDSA, and for more information about kernel storage see "CICS kernel storage" on page 519.

Note: In each of the components of the calculations that follow remember to round their values up to a megabyte boundary.

1. If you would like to specify a generous EDSALIMIT:

   For TRANISO=NO: ECDSA + ERDSA + EUDSA + (64K * MXT)

   For TRANISO=YES: ECDSA + ERDSA + EUDSA + (1MB * MXT)

2. If your current installation EDSAs and MXT values are set to values larger than necessary:

   For TRANISO=NO:

   Peak ECDSA Used + Peak ERDSA Used + (Peak EUDSA Used) - (EUDSA Peak Page Storage in Task Subpools) + (64K * (Peak number of tasks))

   For TRANISO=YES:

   Peak ECDSA Used + Peak ERDSA Used + (Peak EUDSA Used) - (EUDSA Peak Page Storage in Task Subpools) + (1M * (Peak number of tasks))

The minimum EDSALIMIT is 10MB and the default value is 20MB. The maximum EDSALIMIT size is (2 gigabytes - 1 megabyte).

These are guidelines for specifying initial values for the EDSALIMIT. The EDSALIMIT can be dynamically adjusted using the CEMT command without having to stop and restart your CICS system. The safest approach is to:

- Slightly over-specify the EDSALIMIT initially.
- Monitor each EDSA's usage while your system is running near peak loads.
- Tune your EDSALIMIT size using CEMT SET SYSTEM commands.

If you under-specify the EDSALIMIT, your system can go short on storage and it you may not be able to issue CEMT commands to increase the limit.

Dynamic Storage Areas (below the line)

If your installation is constrained for virtual storage below the line, the simplest approach is to set the DSALIMIT equivalent to the sum of the CDSA and UDSA. You will have to consider adjusting these figures so that they use the 256KB limit, see "DSA details" on page 268.

You may find that there is slightly more storage available below the line for DSA storage. CICS pre-allocates approximately 3KB or less of kernel stack storage below the line per task. The majority of kernel stack storage is allocated out of CICS DSAs instead of MVS storage.

DSA details

The DSAs below the line are managed in a similar manner to the EDSAs. The differences in DSA and EDSA management are:

- The extent size for the CDSA, RDSA, and SDSA is in 256KB increments rather than the 1MB size used for the EDSAs.
- If transaction isolation is active, the extent size for the UDSA is 1MB and each UDSA extent must be aligned on a megabyte boundary. If translation isolation is not active, the allocation is in 256KB extents. It is important to keep this in mind because you must allow for some fragmentation between the 256KB extents of the CDSA, RDSA and SDSA compared with the 1 megabyte extents of the UDSA.
- Task storage is 4KB per active task in the UDSA compared with the 1 megabyte or 64KB size for the EUDSA.
- If your applications use more than 4KB per task, you must adjust the formula accordingly (use multiples of 4KB increments if adjusting the formula).
- If your system uses the SDSA and the RDSA, you must allow for these DSAs to be allocated in 256KB increments.

Estimating the DSALIMIT

If you have sufficient virtual storage to adjust your DSALIMIT to a value greater than the sum of your current CDSA + UDSA, the following formulas may be used

Note: In each of the components of the calculations that follow remember to round their values up to a 256KB boundary.

1. If you can afford to specify a generous DSALIMIT:

   CDSA + UDSA + 256K (if both RDSA and SDSA used)

2. If your current installation DSAs and MXT values are set to values larger than necessary:

   Peak CDSA Used + Peak UDSA Used + 256K (if both RDSA and SDSA used)

The minimum DSALIMIT is 2MB and the default value is 5MB. (The maximum DSALIMIT size is 16MB).

As discussed in the EDSALIMIT section, it is safer to slightly over-specify the DSALIMIT than to under-specify it. The DSALIMIT can be tuned to a smaller value after you have obtained data from your running system.

Dynamically altering a DSALIMIT value

Accurate sizing of the DSALIMIT and EDSALIMIT parameters is no longer critical. It is not necessary to recycle your CICS system to make a change to the DSA sizes. CEMT SET SYSTEM, EXEC CICS SET SYSTEM, or CEMT SET DSAS, a new CEMT panel which groups all the storage-related parameters together, can be used to make a change. Care should always be taken, however, when increasing DSALIMIT or EDSALIMIT, as other subsystem problems may occur. For example, an MVS getmain could fail. It is necessary to understand the storage requirement outside the DSAs.

A reduction of DSALIMIT or EDSALIMIT cannot take place if there are no DSA extents free to MVS FREEMAIN. The storage manager will MVS FREEMAIN extent as they become available until the new DSALIMIT or EDSALIMIT value is reached. A short-on-storage condition may occur when reducing DSALIMIT or EDSALIMIT. A new parameter, SOSSTATUS, has been added to CEMT INQUIRE SYSTEM, EXEC CICS INQUIRE SYSTEM, and CEMT INQUIRE DSAS, to give you some indication of short-on-storage conditions.

Removing redundant table entries

Unused table entries increase the cycle requirements because of the additional entries that have to be searched in a table, and thus they tie up real and virtual storage. Elimination of these entries can save some processor resources.

Effects

Entries made in CICS tables, such as unused terminals in the terminal control table (TCT), or unused signons in the signon table (SNT), tie up storage just as if they were used, and can increase the time needed to search through the tables. They can increase CICS startup or restart time as well.

How implemented

Unused entries are most common in the TCT and SNT, but can exist in any CICS table.

How monitored

Use CICS table manager statistics and review those entries with zero counts.

Using modules in the link pack area (LPA/ELPA)

Some CICS management and user modules can be moved into the link pack area (LPA) or the extended link pack area (ELPA). For systems running multiple copies of CICS, this can allow those multiple copies to share the same set of CICS management code.

Effects

The benefits of placing code in the LPA or ELPA are:

- The code is protected from possible corruption by user applications. Because the LPA or ELPA is in protected storage, it is virtually impossible to modify the contents of these programs.
- Performance can be improved and the demand for real storage reduced if you use the LPA or ELPA for program modules. If more than one copy of the same release of CICS is running in multiple address spaces of the same processor, each address space requires access to the CICS nucleus modules. These modules may either be loaded into each of the address spaces or shared in the LPA or ELPA. If they are shared in the LPA or ELPA, this can reduce the working set and therefore, the demand for real storage (paging).
- You can decrease the storage requirement in the private area by judicious allocation of the unused storage in the LPA or ELPA created by rounding to the next segment.

Limitations

Putting modules in the LPA or ELPA requires an IPL of the operating system. Maintenance requirements should also be considered. If test and production systems are sharing LPA or ELPA modules, it may be desirable to run the test system without the LPA or ELPA modules when new maintenance is being tested.

The disadvantage of placing too many modules in the LPA (but not the ELPA) is that it may become excessively large. Because the boundary between the CSA and the private area is on a segment boundary, this means that the boundary may move down one megabyte. The size of the ELPA is not usually a problem.

Recommendations

Use the SMP/E USERMOD called LPAUMOD to select those modules that you want to use for the LPA. This indicates the modules that are eligible for LPA or ELPA. You can use this USERMOD to move the modules into your LPA library.

The objective is to use the LPA wisely to derive the maximum benefit from placing modules in the LPA.

All users with multiple CICS address spaces should put all eligible modules in the ELPA.

How implemented

LPA=YES must be specified in the system initialization table (SIT). Specifying LPA=NO allows you to test a system with new versions of CICS programs (for example, a new release) before moving the code to the production system. The production system can then continue to use modules from the LPA while you are testing the new versions.

An additional control, the PRVMOD system initialization parameter, enables you to exclude particular modules explicitly from use in the LPA.

For information on installing modules in the LPA, see the *CICS/ESA Installation Guide*.

Map alignment

CICS maps that are used by basic mapping support (BMS) can be defined as aligned or unaligned. In aligned maps, the length field associated with a BMS data field in the BMS DSECT is always aligned on a halfword boundary. In unaligned maps, the length field follows on immediately from the preceding data field in the map DSECT.

A combination of aligned and unaligned maps can be used.

Effects

In unaligned maps, there is no guarantee that the length fields in the BMS DSECT are halfword-aligned. Some COBOL and PL/I compilers, in this case, generate extra code in the program, copying the contents of any such length field to, or from, a halfword-aligned work area when its contents are referenced or changed.

Specifying map alignment removes this overhead in the application program but increases the size of the BMS DSECT, at worst by one padding byte per map data field, and marginally increases the internal pathlength of BMS in processing the map. The best approach, therefore, is to use unaligned maps, except where the compiler being used would generate inefficient application program code.

In COBOL, an unsynchronized map generates an unsynchronized structure. In PL/I, an unaligned map generates a map DSECT definition as an unaligned structure. Correspondingly, aligned maps produce synchronized structures in COBOL and aligned structures in PL/I.

Some of the VS COBOL compilers have an option that does not generate the extra copy statements associated with an unsynchronized structure, but other COBOL compilers do. If this option is available, it should be specified because you do not then need aligned maps.

Limitations

In CICS, BMS maps are always generated in groups (“map sets”). An entire map set must be defined as aligned or unaligned. Also, maps may be used by application programs written in a variety of languages. In these cases, it is important to choose the option that best suits the combination of programs and, if there is any requirement for both aligned and unaligned maps, the ALIGNED option should be taken.

Conversion of maps from aligned to unaligned or conversely should be avoided if possible, because changing the map DSECT also requires reassembly or recompilation of all application programs that reference it.

How implemented

Map alignment is defined when maps are assembled. Aligned maps use the SYSPARM(A) option. The BMS=ALIGN/UNALIGN system initialization parameter defines which type of map is being used.

The map and map set alignment option can also be specified when maps and map sets are defined using the screen definition facility (SDF II) licensed program product. For more information, see the *Screen Definition Facility II Primer for CICS/BMS Programs*.

How monitored

The importance of map alignment may be found by inspecting programs that handle screens with a large number of fields. Try recompiling the program when the BMS DSECT is generated first without, and then with, the map alignment option. If the program size, as indicated in the linkage edit map, drops significantly in the second case, it is reasonable to assume there is high overhead for the unaligned maps, and aligned maps should be used if possible.

Resident, nonresident, and transient programs

Programs, map sets, and partitions can be defined as RESIDENT(NO|YES) and USAGE(NORMAL|TRANSIENT). Programs can be defined as RELOAD(NO|YES).

Effects

| Any program defined in the CSD is loaded into the CDSA, RDSA, SDSA, ECDSA, ERDSA, or ESDSA on first usage. RELOAD(YES) programs cannot be shared or reused. A program with RELOAD(YES) defined is only removed following an explicit EXEC CICS FREEMAIN. USAGE(TRANSIENT) programs can be shared, but are deleted when the use count falls to zero. RESIDENT(NO) programs become eligible for deletion when the use count falls to zero. The CICS loader domain progressively deletes these programs as DSA storage becomes shorter, on a least-recently-used basis.

RESIDENT(YES) programs are not normally deleted. If NEWCOPY is executed for any program, a new copy is loaded and used on the next reference and the old copy becomes eligible for deletion when its use count falls to zero.

On a CICS WARM start, an initial free area for the various resident program subpools is allocated. The size of this area is based on the total lengths of all currently loaded resident programs as recorded during the preceding CICS shutdown. When a resident program is loaded, CICS attempts to fit it into the initial free area. If it does not fit, it is loaded outside the initial free area, and the space inside the initial free area remains unallocated until other (smaller) resident programs are loaded into it. This could occur if a resident program has increased its size since it was last loaded (before the preceding CICS shutdown). If the program in question is very large, storage problems could arise due to the large amount of unused storage in the initial free area allocated for resident programs.

Recommendations

Because programs that are not in use are deleted on a least-recently-used (LRU) basis, they should be defined as RESIDENT(NO) unless there are particular reasons to favor particular programs by keeping them permanently resident. Variations in program usage over time are automatically taken account of by the LRU algorithm.

Thus, a much-used nonresident program is likely to remain resident anyway, whereas – during periods of light usage – a resident program could be wasting the virtual storage it permanently occupies.

For programs written to run above the 16MB line, you should be able to specify EDSALIM large enough such that virtual storage is not a constraint.

If a program is very large or frequently updated such that its size increases, consider defining it as non-resident and issuing a LOAD with the HOLD option as part of PLTPI processing. The program will not be released during program compression, but also ensures that there will not be a significant amount of initial free storage reserved for resident programs which may go unused because the new (larger) program will not fit into it.

The reasons for defining a program as RESIDENT might be:

- Possible avoidance of storage fragmentation, because all such programs are in a single block of storage (but not new copies of programs).
- Programs are needed to deal with potential crises (for example, CEMT).
- Heavy contention on the DFHRPL program libraries. However, this should usually be dealt with by data set placement or other DASD tuning, or use of MVS library lookaside to maintain program copies in an MVS dataspace. See "Use of LLA (MVS/ESA library lookaside)" on page 189.

How monitored

The tuning objective is to optimize throughput at an acceptable response time by minimizing virtual storage constraint. There are specific loader domain statistics for each program.

Putting application programs above the 16MB line

CICS/ESA 4.1 keeps RMODE(ANY) application programs in the EDSA, which is in MVS extended virtual storage above the 16MB line. Work areas associated with the programs may also reside above the 16MB line.

Effects

It is possible to LINK or XCTL between 31-bit mode programs and 24-bit mode programs. You can convert programs to 31-bit mode programs and move them above the 16MB line to the extended private area. Moving programs above the 16MB line frees that amount of virtual storage below the 16MB line for other use.

See the *CICS/ESA Operations and Utilities Guide* for information on using programs from the LPA or extended link pack area (ELPA).

Using the ELPA is usually better than using the extended private area when multiple address spaces are employed, because the program is already loaded when CICS needs it, and real-storage usage is minimized.

| When running a CICS system with transaction isolation enabled, performance
| benefits can be gained by moving transactions and application programs above the
| line. Program work areas are then obtained from the EUDSA with a MB pagesize
| rather than the UDSA which has a 4KB pagesize.

Where useful

This facility is useful where there is demand for virtual storage up to the 16MB line and there is sufficient real storage.

Limitations

Some program languages do not support programs that run above the 16MB line. Assembler H Version 2, VS COBOL II, and PL/I Release 5.1 are examples of languages that do provide the necessary support.

Because the purpose of using virtual storage above the 16MB line is to make the space below this available for other purposes, there is an overall increase in the demand for real storage when programs are moved above the 16MB line.

There is a restriction on the use of COMMAREAs being passed between programs running in 31-bit addressing mode and programs running in 24-bit addressing mode. COMMAREAs passed from a 31-bit program to a 24-bit program must be capable of being processed by the 24-bit program, therefore they must not contain 31-bit addresses: addresses of areas that are themselves above the 16MB line.

How implemented

Programs that are to reside above the 16MB line must be link-edited with the AMODE(31),RMODE(ANY) options on the MODE statement of the link-edit. See the *CICS/ESA Operations and Utilities Guide* for further information.

Transaction isolation and real storage requirements

When using transaction isolation there is a cost in terms of real storage. Paging problems can result if insufficient real storage is allocated, which then affects performance. The cost is very much based on the number of subspaces in use in the system, and the size of EDSALIM.

Since the pagesize of the EUDSA is one MB, EDSALIM is likely to be very large for a CICS system which has transaction isolation active. Since this virtual storage needs to be mapped with page and segment tables using real storage, an increase in the real storage usage can occur. In addition to the real storage used to map the virtual storage for the EDSALIMIT, subspaces also require real storage. For example:

- Each subspace requires 2.5 pages.
- Assuming each transaction in the system requires a unique subspace. (transaction definition TASKDATAKEY(USER) and ISOLATE(YES)). Real storage required is MXT * 2.5 pages.
- If each transaction in the system requires a page of storage in the EUDSA (1MB page), a page table is required to map the storage. Real storage is MXT * 1 page.
- A further three pages are required to give a total of Real storage = MXT * (1 + 2.5 pages) + 3 pages.
- All of this real storage is allocated from the ELSQA.

The figures for the real storage usage is in addition to that required for a CICS/ESA system which does not have transaction isolation active.

Note: Where a page means a 4KB page of real storage.

Limiting the expansion of subpool 229 by using VTAM pacing (PACING, VPACING)

Subpool 229 may be expanded if batch type terminals send data faster than a CICS transaction can process that data. The use of secondary to primary pacing, sometimes called inbound pacing, limits the amount of data queued in subpool 229 for any given batch terminal.

PACING controls the flow of traffic from the network control program (NCP) to the terminal and does not affect the processor activity as such. VPACING on the other hand controls the flow of traffic between the host and the NCP.

The VPACING parameter of the CICS APPL statement determines how many messages can be sent in a session to the VTAM application program by another VTAM logical unit without requiring that an acknowledgment (called a “pacing response”) be returned. The host sends data path information units (PIUs) according to the definition of VPACING. The first PIU in a group carries a pacing indicator in the RH. When this PIU is processed by the NCP, the NCP sends a response to the host with the same pacing indicator set to request a new pacing group. This means that, for every **x** PIUs to a terminal and every **y** PIUs to a printer, the pacing response traffic must flow from the NCP to the host which, based on the volume of traffic, could cause a significant increase in host activity.

Normally, VPACING is implemented when a shortage of NCP buffers requires controlling the volume of flow between the host and the NCP. You may be able to lessen the effect on the processor by increasing the VPACING value to what the NCP can actually tolerate.

The PACING parameter is required for most printers, to match the buffer capacity with the speed of printing the received data. Terminals do not normally require pacing unless there is a requirement to limit huge amounts of data to one LU, as is the case with some graphics applications. Use of pacing to terminals causes response time degradation. The combination of PACING and VPACING causes both response time degradation and increased processor activity, and increased network traffic.

Recommendations

PACING and VPACING should be specified for all terminals to prevent a “runaway” transaction from flooding the VTAM network with messages and requiring large amounts of buffer storage. If a transaction loops while issuing SENDs to a terminal, IOBUF (CSA storage) and NCP buffers may fill up causing slowdowns and CSA shortage conditions.

PACING and VPACING should always be specified high enough so that normal data traffic may flow without being regulated, but excessive amounts of data are prevented from entering the network and impairing normal data flow.

How implemented

For secondary to primary pacing, you must code:

- SSNDPAC=nonzero value in the LOGMODE entry pointed to by the secondary application program
- VPACING=nonzero value on the APPL definition for the secondary application.

The value used is coded on the VPACING parameter. If either of these values are zero, no pacing occurs.

Specify VPACING on the APPL statement defining the CICS region, and any nonzero value for the SSNDPAC parameter on the LU statement defining the batch device. You should ensure that the device supports this form of pacing by referring to the component description manual for that device.

For further information on the selection criteria for values for the PACING and VPACING parameters, see the *ACF/VTAM Version 2 Planning and Installation Reference* manual.

Dynamic log buffer size (DBUFSZ)

The dynamic log buffer stores backout information in the dynamic log for dynamic transaction backout purposes. Dynamic transaction backout is used to reverse changes made to recoverable resources in CICS if a transaction abend or syncpoint rollback occurs.

The dynamic log is allocated in the dynamic storage area above the 16MB line (ECDSA). The minimum allocation is 500 bytes.

The dynamic log buffer is not acquired until a recoverable resource has been modified and the first dynamic log record for the buffer is to be written.

The actual buffer size allocated is not the same for all transactions, but is dynamically self-tuned for each transaction to a value that is just large enough to accommodate the log almost all of the time. The (occasional) requirements exceeding the allocation are satisfied by spilling to a chain of main storage allocations in the ECDSA.

Effects

The system initialization parameter, DBUFSZ, influences the self-tuning algorithm by constraining the dynamic log buffer size to not more than DBUFSZ, and defining the initial allocation for each transaction as half of DBUFSZ.

Where useful

Dynamic transaction backout is used by any transaction that updates recoverable resources. DBUFSZ applies to any such transaction.

Dynamic transaction backout is automatically provided, but it has no effect (that is, no dynamic log is acquired) for a task unless it issues a request to update recoverable resources.

Limitations

Because the initial allocation of space for the dynamic log buffer is half the maximum specified in the DBUFSZ system initialization parameter, the dynamic self-tuning mechanism may take some time to settle down. Statistics collected over short periods could be misleading.

Recommendations

Normally, little use of overflow storage is required, and this should be achieved automatically if the DBUFSZ system initialization parameter is large enough.

Look at the dynamic transaction backout statistics. If the ratio of the number of records logged and the number of records spilled is higher than 2%, consider increasing the DBUFSZ value in the SIT.

How implemented

Dynamic transaction backout is automatically available in CICS with resource definition online (RDO). The maximum size of the dynamic log buffer is defined by the system initialization parameter, DBUFSZ.

How monitored

The CICS dynamic transaction backout statistics give information on dynamic buffer activity, and the number of records spilled.

Chapter 22. MRO and ISC/IRC

This chapter includes the following topics:

CICS intercommunication facilities

CICS intercommunication facilities allow different CICS systems to communicate and share resources with each other. These facilities consist of the following components:

- Function shipping
- Distributed transaction processing
- Asynchronous processing
- Transaction routing.
- Distributed program link.

For details of the CICS intercommunication facilities, see the *CICS/ESA Intercommunication Guide*. See also “Splitting online systems: virtual storage” on page 256, and “Splitting online systems: availability” on page 181.

If there are a number of intercommunication requests for each transaction, function shipping generally incurs the most overhead. The number of requests per transaction that constitutes the break-even point depends on the nature of the requests.

Both distributed transaction processing (DTP) and asynchronous processing are, in many cases, the most efficient method of intercommunication because a variety of requests can be batched in one exchange. DTP, however, requires an application program specifically designed to use this facility. For information about designing and developing DTP, see the *CICS/ESA Distributed Transaction Programming Guide*.

Transaction routing, in most cases, involves one input and one output between systems, and the overhead is minimal.

Multiregion operation (MRO), in general, causes less processor overhead than intersystem communication (ISC) because the SVC pathlength is shorter than that through the multisystem networking facilities of VTAM. This is particularly true with CICS MRO, which provides a long-running mirror transaction and fastpath transformer program.

Some SVC-processing overhead can be eliminated from MRO in CICS with the use of MVS cross-memory services. Cross-memory services use the MVS common system area (CSA) storage for control blocks, not for data transfer. This can also be of benefit. Note, however, that MVS requires that an address space using cross-memory services be non-swappable.

For situations where ISC is used across MVS images, consider using XCF/MRO. XCF/MRO consumes less processor overhead than ISC.

ISC mirror transactions can be prioritized. The CSMI transaction is for data set requests, CSM1 is for communication with IMS/ESA systems, CSM2 is for interval control, CSM3 is for transient data and temporary storage, and CSM5 is for IMS/ESA DB requests. If one of these functions is particularly important, it can be prioritized above the rest. This prioritization is not effective with MRO because any attached mirror transaction services any MRO request while it is attached.

If ISC facilities tend to flood a system, this can be controlled with the VTAM VPACING facility. Specifying multiple sessions (VTAM parallel sessions) increases throughput by allowing multiple paths between the systems.

CICS also allows you to specify a VTAM class of service (COS) table with LU6.2 sessions, which can prioritize ISC traffic in a network. Compare the performance of CICS function shipping with that of IMS/ESA data sharing.

Limitations

- Use of intercommunication entails trade-offs as described in “Splitting online systems: virtual storage” on page 256 and “Splitting online systems: availability” on page 181.
- Increased numbers of sessions can minimally increase real and virtual storage but reduce task life. The probable overall effect is to save storage.
- MVS cross-memory services reduce CSA and cycle requirements.
- MRO high performance facilities reduce processing requirements.
- IMS/ESA data sharing usually reduces processor requirements.
- Accessing DL/I databases via the IMS DBCTL facility reduces processor requirements relative to function shipping.
- For MRO considerations, read about the secondary effects of the region exit interval (ICV) on page “Region exit interval (ICV)” on page 187.

How implemented

See the *CICS/ESA Installation Guide* for information about resetting the system for MRO or ISC. See also “Splitting online systems: virtual storage” on page 256.

How monitored

CICS ICS/IRC statistics (see page 361) show the frequency of use of intercommunication sessions and mirror transactions. The VTAM trace, an SVC trace, and RMF give additional information.

Intersystems Session Queue Management

When intersystems links are added to the system there is the possibility that they cannot respond adequately to transaction requests because the remote system is performing badly. The poor performance can be due either to a long-term condition such as lack of resource or overloading, or a temporary situation such as a dump being taken. In any case there is the danger that the problem can cause a long queue to form in the requesting system.

Mechanisms are provided in CICS for:

- Protection of the requesting system from using too many resources whilst transactions queue for the use of the intersystems sessions.
- Detection of problems in remote systems. CICS can issue messages to indicate a problem on an intersystems connection and the parameters control the criteria that are used to determine when a problem exits, or has gone away.

The two mechanisms are:

1. The QUEUELIMIT and MAXQTIME parameters on the connection resource definition.

   The QUEUELIMIT parameter limits the number of transactions which can be queued in allocate processing waiting for a session to become free. Any transactions which try to join a queue already at its limit are rejected.

   The MAXQTIME parameter is a control on the wait time of queued allocate requests that are waiting for free sessions on a connection that appears to be unresponsive. If the rate of processing of the queue indicates that a new allocate will take longer than the specified time to reach the head of the queue, the whole queue is purged.

2. The XZIQUE user exit, which is given control when an allocate request is about to be queued, or the first time it succeeds after a suspected problem. The XZIQUE exit can control the queue in the same way as the CEDA parameters, or you can use it to add more sophisticated controls of your own.

Both mechanisms produce the same effect on the application program which issued the allocate; a SYSIDERR condition is returned. Return codes are also provided to the dynamic routing program to indicate the state of the queue of allocate requests.

CICS/ESA Resource Definition Guide contains more description of the CEDA commands; and the *CICS/ESA Customization Guide* gives programming information about the XZIQUE exit and its relationship with the rest of CICS, including application programs and the dynamic routing program.

Relevant statistics

For each connection CICS records the following:

- The number of allocates queued for the connection, and the peak value of this number. (`Peak outstanding allocates` in the Connection statistics.)

  You can use this statistic to see how much queuing normally takes place on connections in your system. If there is occasionally a large queue you should consider controlling it. “Are there enough sessions defined?” on page 59 has more advice on setting the right number of sessions for your connections.

For each of the queue control mechanisms CICS records the following statistics for each connection:

- The number of allocates which were rejected due to the queue becoming too large
- The number of times the queue was purged because the throughput was too slow
- The number of allocates purged due to slow throughput.

"ISC/IRC system and mode entry statistics" on page 56 also contains an explanation of these, and other connection statistics.

Ways of approaching the problem and recommendations

The queue limit mechanism should be used to control the number of tasks waiting for the use of an intersystems link. You should use the control to ensure that even at its maximum length the queue does not use too many, and certainly not all, of the MXT slots in the system. You can also use the MAXACTIVE setting of a TRANCLASS definition to do this if you can segregate your transactions into classes which correspond to the remote regions they require.

You should allow sufficient intersystems sessions to enable their free availability during normal running. Session definitions do not occupy excessive storage, and the occupancy of transaction storage probably outweighs the extra storage for the session. The number of sessions should correspond to the peak number of transactions in the system which are likely to use the connection - you can see the maximum number of sessions being used from the terminal statistics for the connection. If all sessions were used the connections statistics show the number of times allocates were queued compared with the total number of requests.

Even in a system which has no problems there are significant variations in the numbers of transactions which are active at any time, and the actual peak number may be larger than the average over a few minutes at the peak time for your system. You should use the average rather than the actual peak; the queueing mechanism is intended to cope with short term variations, and the existence of a queue for a short time is not a cause for concern.

The start of a queue is used by the queue limiting mechanism as a signal to start monitoring the response rate of the connection. If queues never form until there is a big problem the detection mechanism is insensitive. If there are always queues in the system it will be prone to false diagnosis.

You should set the queue limit to a number which is roughly the same size as the number of sessions - within the limits imposed by MXT if there are many connections whose cumulative queue capacity would reach MXT. In this latter case you might need to design your own method - using ZXIQUE - of controlling queue lengths so that the allocation of queue slots to connections was more dynamic.

You should set the MAXQTIME parameter with regard to the time you think the users of the system should be prepared to wait for a response in the case of a potential problem, but bear in mind that you should not set it to be short in combination with a queue limit that is low, since this leads to a very sensitive detection criterion.

Monitoring the settings

The number of allocates rejected by the queue control mechanism should be monitored. If there are too many it may indicate a lack of resources to satisfy the demands on the system - or poor tuning.

The number of times the queue is purged should indicate the number of times a serious problem occurred on the remote system. If the purges do not happen when the remote system fails to respond then examine the setting of the MAXQTIME parameter - it may be too high, and insensitive. If the indication of a problem is too frequent and causes false alarms simply due to variations in response time of the remote system then the parameter may be too low, or the QUEUELIMIT value too low.

Terminal input/output area (SESSIONS IOAREALEN) for MRO sessions

For MRO function shipping, the SESSIONS definition keyword, IOAREALEN, is used. This keyword regulates the length of the terminal input/output area (TIOA) to be used for processing messages transmitted on the MRO link. These TIOAs are located above the 16MB line.

Effects

The IOAREALEN value controls the length of the TIOA which is used to build a message transmitted to the other CICS system (that is, an outgoing message).

Two values (value1 and value2) can be specified. Value1 specifies the initial size of the TIOA to be used in each session defined for the MRO connection. If the size of the message exceeds value1, CICS acquires a larger TIOA to accommodate the message.

Only one value is required, however if value2 is specified CICS will use value2 whenever the message cannot be accommodated by the value1.

A value of zero causes CICS to get a storage area exactly the size of the outgoing message, plus 24 bytes for CICS requirements.

If the IOAREALEN value is not specified, it defaults to 4KB.

Where useful

The IOAREALEN keyword can be used in the definition of sessions for either MRO transaction routing or function shipping. In the case of MRO transaction routing, the value determines the initial size of the TIOA, whereas the value presents some tuning opportunities in the MRO function shipping environment.

Limitations

Real and virtual storage can be wasted if the IOAREALEN value is too large for most messages transmitted on your MRO link. If IOAREALEN is smaller than most messages, or zero, excessive FREEMAIN and GETMAIN requests can occur, resulting in additional processor requirements.

Recommendations

For optimum storage and processor utilization, IOAREALEN should be made slightly larger than the length of the most commonly encountered formatted application data transmitted across the MRO link for which the sessions are defined. For efficient operating system paging, add 24 bytes for CICS requirements and round the total up to a multiple of 64 bytes. A multiple of 64 bytes (or less) minus 24 bytes for CICS requirements ensures a good use of operating system pages.

How implemented

The TIOA size can be specified in the IOAREALEN keyword of the SESSIONS definition.

Batching requests (MROBTCH)

Certain events in a region can be accumulated in a batch prior to posting, until the number specified in the MROBTCH system initialization parameter is reached (or ICV times out). Then, the region is started so that it can process the requests. The batching of MRO requests includes some non-MRO events such as:

- VSAM physical I/O completion
- Subtasked (mostly VSAM) request completion (if SUBTSKS=1 is specified)
- DL/I request completion implemented through DBCTL.

Strictly speaking, batching is applicable to a TCB rather than the region. MROBTCH is applied only to the ‘quasi-reentrant’ mode TCB.

Effects

Compared to no batching (MROBTCH=1, that is, the default), setting MROBTCH=n has the following effects:

- Up to [(n–1)*100/n]% saving in the processor usage for waiting and posting of that TCB. Thus, for n=2, 50% savings may be achieved, for n=3, 66% savings, for n=6, 83% savings, and so on.
- An average cost of (n+1)/2 times the average arrival time for each request actually batched.
- Increased response time may cause an increase in overall virtual storage usage as the average number of concurrent transactions increases.
- In heavily loaded systems at peak usage, some batching can happen as a natural consequence of queueing for a busy resource. Using a low MROBTCH value greater than one may then decrease any difference between peak and off-peak response times.

Setting MROBTCH higher than 6 is not recommended as the decreasing additional processor saving is unlikely to be worth the further increased response time.

You require a relatively low value of MROBTCH for ICV to maintain reasonable response time during periods of low utilization.

Recommendations

Depending on the amount of response time degradation you can afford, you can set MROBTCH to different values using either CEMT or EXEC CICS SET SYSTEM MROBATCH(value).

The recommendation is to use CEMT or EXEC CICS INQUIRE SYSTEM MROBATCH(value) to arrive at a suitable batch value for a given workload. Then use CEMT or EXEC CICS SET SYSTEM MROBATCH(value) to reset the figure appropriately. See the *CICS/ESA CICS-Supplied Transactions* manual for more information about CEMT; for programming information about the EXEC CICS system programming commands, see the *CICS/ESA System Programming Reference*.

During slow periods the ICV unconditionally dispatches the region, even if the batch is not complete and provides a minimum delay. In this case, set ICV to 500 milliseconds in each region.

Extending the life of mirror transactions (MROLRM and MROFSE)

This MROLRM system initialization parameter can have a significant effect on the performance of a workload in an MRO function shipping environment.

Setting **MROLRM=NO** causes the mirror to be attached and detached for each function-shipped request until the first request for a recoverable resource or a file control start browse is received. After such a request is received, the mirror remains attached to the session until the calling transaction reaches syncpoint.

Setting **MROLRM=YES** in a region receiving function shipping requests causes a mirror transaction to remain attached to the MRO session from first request until the calling transaction reaches syncpoint. This option causes system-dependent effects, as follows:

- Some systems show significant improvements in processor utilization per transaction. They are likely to be systems with a significant percentage of inquiry transactions, each with multiple VSAM calls, or transactions with many reads followed by a few updates.
- Some systems show no performance difference. Workloads using IMS/ESA, or transactions that make a lot of use of VSAM-update or browse-activity, may fall into this category.
- Some systems could be degraded because there is an extra flow at syncpoint. An example of this would be a system with a very simple inquiry transaction workload.

In general, setting MROLRM=YES is recommended.

Setting MROFSE=YES in a region issuing function shipping requests causes the mirror to remain attached to the MRO session from the first request until the calling transaction reaches end of task. (A DPL flow will cause the session to be freed at the following syncpoint.) This parameter should be used in conjunction with MROLRM=YES on the system receiving the function shipping requests.

- Some systems show significant improvements in processor utilization for each transaction. They are likely to be systems that benefit from MROLRM=YES

and have multiple syncpoints with function shipping requests issued between each syncpoint.

- Some systems could be degraded because:
  - the session may be held until the end of the task rather than to the user syncpoint.
  - there may be redundant syncpoint flows if a significant number of the units of work have no function shipping.

It may be necessary to increase system initialization parameter MXT because of the extended life of the mirror task.

It may also be necessary to increase the number of SEND sessions defined to the front end system with a consequential change to the number of RECEIVE sessions defined on the back end. MROLRM=YES implies an increase in the number of RECEIVE sessions defined to the back end system.

Deletion of shipped terminal definitions (DSHIPINT and DSHIPIDL)

In a transaction routing environment, terminal definitions can be "shipped" from a terminal-owning region (TOR) to an application-owning region (AOR). A shipped terminal definition in an AOR becomes redundant when:

- The terminal user logs off.
- The terminal user stops using transactions which route to the AOR.
- The TOR on which the user is signed on is shut down.
- The TOR is restarted without recovering autoinstalled terminal definitions, and the autoinstall user program DFHZATDX assigns a new set of terminal ids to the same set of terminals.

Shipped terminal definitions which have become redundant may need to be deleted. Long-lasting shipped terminal definitions do not generally cause storage problems because of the relatively small amounts of storage which they occupy. However, there are other considerations, such as security, which may require that redundant shipped terminal definitions are not allowed to persist in an AOR.

The CICS-supplied transaction CRMF periodically scans the shipped terminal definitions in the AOR and flags those which it has determined to be redundant. If any redundant definitions have been identified, then the CICS-supplied transaction CRMD is invoked to delete them. This processing is referred to as the CICS timeout delete mechanism.

The system initialization parameters DSHIPINT and DSHIPIDL control the amount of time for which a redundant shipped terminal definition is allowed to survive and the frequency at which shipped terminal definitions are tested for redundancy.

Effects

The DSHIPIDL system initialization parameter determines the period of time for which a shipped terminal definition is allowed to remain inactive before it may be flagged for deletion. The DSHIPINT system initialization parameter determines the time interval between invocations of the CRMF transaction. CRMF will examine all shipped terminal definitions to determine which of them have been idle for longer

than the time interval specified by DSHIPIDL. If CRMF identifies any redundant terminal definitions, it will invoke CRMD to delete them.

Where useful

The CRMF/CRMD processing will be most effective in a transaction routing environment in which there may be shipped terminal definitions in an AOR which remain idle for considerable lengths of time.

Limitations

Once CRMF/CRMD processing has deleted a shipped terminal definition, the terminal definition will need to be re-shipped when the terminal user next routes a transaction from the TOR to the AOR. Care should therefore be taken not to set DSHIPIDL to a value which is low enough to cause shipped terminal definitions to be frequently deleted between transactions. Such processing could incur CPU processing costs, not just for the deletion of the shipped terminal definition, but also for the subsequent re-installation when the next transaction is routed.

Bear in mind that if a large value is chosen for DSHIPINT, it will influence the length of time that a shipped terminal definition will survive. The period of time for which a shipped terminal definition will remain idle before being deleted will be extended by an average of half of the DSHIPINT value. This is because once the terminal has exceeded the limit for idle terminals set by the DSHIPIDL parameter, it will have to wait (for an average of half of the DSHIPINT interval) before CRMF is scheduled to identify the terminal definition as idle and flag it for CRMD to delete. When the DSHIPINT interval is significantly longer than the DSHIPIDL interval (which will be the case if the default values of 120000 for DSHIPINT and 020000 for DSHIPIDL are accepted), DSHIPINT becomes the dominant factor in determining how long an idle shipped terminal definition will survive before being deleted.

Recommendations

Ensure that unnecessary processing is not being generated by too low a value being assigned to DSHIPIDL. The storage occupied by the shipped terminal definitions is not normally a concern, so the default value which specifies a maximum idle time of 2 hours is reasonable, unless other concerns (such as security) suggest that it should be shorter.

Decide whether you wish to delete idle shipped terminal definitions incrementally or en masse. CRMF processing in itself causes negligible CPU overhead, so a low value for DSHIPINT may therefore be specified at little cost, providing that a sensible value for DSHIPIDL has been chosen. Specifying a low value for DSHIPINT so that CRMF runs relatively frequently could mean that idle terminal definitions are identified in smaller batches, so that CRMD processing required to delete them is spread out over time.

A higher value for DSHIPINT, especially if the default value of 12 hours is accepted, may mean that when CRMF runs it identifies a considerable number of idle terminal definitions, so that a larger burst of CPU is required for the CRMD processing. You may wish to ensure that this type of processing occurs during periods of low activity in the CICS region. To this end, the CEMT INQUIRE/SET/PERFORM DELETSHIPPED commands (and their equivalent

CICS/ESA SPI commands) are available to help you schedule when the CRMF transaction will be invoked.

How implemented

The maximum length of time for which a shipped terminal definition is allowed to remain idle before it may be flagged for deletion is specified by the CICS system initialization parameter DSHIPIDL. The interval between scans to test for idle definitions is specified by the CICS system initialization parameter DSHIPINT.

Both these parameters may be adjusted by the CEMT INQUIRE/SET DELETSHIPPED commands. Note that the revised interval to the next invocation of the timeout delete mechanism starts from the time the command is issued, not from the time of the last invocation, nor from the time of CICS startup.

The timeout delete mechanism may be invoked immediately by the CEMT PERFORM DELETSHIPPED command or its CICS/ESA SPI equivalent.

How monitored

The CICS terminal autoinstall statistics provide information on the current setting of the DSHIPINT and DSHIPIDL parameters, the number of shipped terminal definitions built and deleted, and the idle time of the shipped terminal definitions.

Chapter 23. Programming considerations

This chapter includes the following topics:

BMS map suffixing and the device-dependent suffix option

CICS BMS allows you to use different versions of a map set for different device types by specifying a one-letter suffix for each different device type. Use of this facility requires the BMS device-dependent suffix (DDS) option. For further information on map set suffixes, see the *CICS/ESA Application Programming Guide*.

Where only one version of a map is involved, it is optional whether the device type suffix is coded. If the DDS option is being used, it is more efficient to use the device suffixes than to leave the suffix blank. This is because, if the DDS option applies, CICS first looks for a map set with a suffix name and then searches again for a map with a blank suffix. Processor cycle requirements are reduced by eliminating the second table lookup.

Effects

If only one device type is used with all maps in a CICS system and all devices have the same screen size, CICS can be initialized to look for a blank suffix, thus eliminating the second lookup.

If the map is to be used with multiple devices, multiple maps with the same basic source are needed because the device type needs to be specified, and suffixing is required in this case.

Recommendation

If you decide that you need device-dependent suffixing, you should suffix all your map sets. If you do not need it, use blank suffixes (no suffix at all) and specify the NODDS option in BMS.

How implemented

Maps are named in the link-edit process. These names are defined in the MAPSET definition. Specifying NODDS in the BMS= system initialization parameter determines that map suffixing is not used in CICS.

How monitored

No direct measurement of map suffixing is given.

COBOL RESIDENT option

Compiling online COBOL programs with the optional RESIDENT and mandatory NODYNAM compiler options (available with the VS COBOL Version 4 and VS COBOL II compilers) allows those application programs to share common COBOL library subroutines. The library subroutines are not compiled into each module; instead, linkage is set up to a common set of subroutines and so only one copy of each module exists in a system rather than one copy for each program.

Effects

CICS COBOL programs may use either the RESIDENT or NORESIDENT compiler options. In either case, NODYNAM must also be specified.

Specifying NORESIDENT means that any COBOL library subroutines required by a COBOL program (for example, to execute an EXAMINE operation) should be link-edited with the user's code. If any subprograms are link-edited with the main program, these may also require library subroutines; one copy of each required routine is included in the load module, no matter how many places it is called from.

Using NORESIDENT means that each COBOL program used in CICS includes its own set of required library subroutines. How much bigger this makes each program depends on the way that it is written. It is likely that many programs need a common set of COBOL library routines, and each such program includes its own copy of the set. The storage cost of this depends on how many programs are present simultaneously in virtual storage: how many are permanently resident, how many are loaded into CICS dynamic storage at any time, and what the extra storage cost per program is.

Under the RESIDENT option, any library routines required by the COBOL program are not link-edited. Instead, COBOL initialization code tries to locate them when the program (or subprogram) is first invoked, by issuing MVS LOAD macros. If the library routines have been set up in the link pack area (LPA), MVS simply passes the routine addresses to COBOL, which then inserts them into the COBOL program, without any extra I/O. But if the routines are not in the LPA, they have to be loaded from the disk libraries (STEPLIB, JOBLIB, and LINKLIB).

The COBOL execution-time routines for programs compiled with RES,NODYNAM are loaded with an MVS LOAD macro (SVC) only if they are not already loaded.

When a COBOL program is initialized, it unconditionally loads ILBCBL00, which is loaded only the first time it is referenced. This module has a vector table of module names and addresses. When a module is needed, a search is made to check whether that module is already loaded. If it has been loaded previously, its address is used. If the module that is referenced has not been loaded, it is loaded by MVS and its address is placed in the vector table for future reuse.

These modules are loaded from the LINKLIB, any LINKLST data set, JOBLIB, STEPLIB, or perhaps from the LPA, but ***not*** from DFHRPL (COBOL does not know anything about DFHRPL). The MVS LOAD is not issued with any DCB parameter, so only the standard MVS LOAD hierarchy can be used. They should not be

defined on the CSD, because this is not known to COBOL either. RES,NODYNAM is a COBOL feature, ***not*** a CICS feature.

In the CICS environment, many of the required COBOL library routines can be placed in the LPA; this gives an environment similar to that of the PL/I shared library. This reduces the size of each COBOL program according to its subroutine requirement. The saving in space normally offsets any extra space needed for one copy of each library subroutine in the LPA (assuming the modules have not already been put in the LPA for use by batch COBOL programs).

ILBCBL00 must ***not*** be in the LPA, however. It is loaded into the user's region and has a set of vector addresses for the other modules that may be used by the COBOL programs within that region. When a new module is requested, its address is not known and the COBOL interface routine loads that module and places its address in the list to be used again, as explained above.

If the COBOL RESIDENT option is used under CICS and the desired COBOL subroutines are not LPA-resident, an MVS LOAD is issued for each such subroutine when the first COBOL CICS program to require that subroutine is initialized. This causes a synchronous halt to CICS until the I/O is complete.

Using the RESIDENT option saves real and virtual storage. Approximately 3.5KB of storage can be saved per program, depending on the release of COBOL used and what subroutines are referenced. These subroutines are loaded the first time they are referenced, or can reside in the LPA and be shared by all programs that reference them.

Note that the requirement is that the ILBxxxxx routines be reentrant, not the application code. This feature is purely a COBOL feature and CICS does not have any specific code to support it.

The ability of CICS to share code between multiple concurrent users is based on pseudoreentrance. This means that the program must be reentrant at the time you pass control to CICS with a command but not between times.

Limitations

Recompilation of programs is required for programs not using these options.

Recommendations

Ensure that the resident subroutines are placed in the link pack area so that an MVS LOAD is not incurred the first time they're referenced.

How implemented

Resident subroutines are implemented by specifying RESIDENT and NODYNAM when the program is compiled.

How monitored

A link-edit map shows storage savings. RMF shows overall real and virtual storage usage.

PL/I shared library

The PL/I optimizing compiler has a facility whereby those resident library modules likely to be used in more than one program simultaneously can be stored together in the link pack area, from where they can be invoked from any region. This facility, known as the **PL/I shared library**, is available to PL/I programs running as CICS applications, provided that they were compiled by the PL/I optimizing compiler. The PL/I shared library is another facility that helps the user to conserve storage.

PL/I resident library routines can be shared between multiple CICS PL/I programs, rather than being compiled into each separate PL/I application program. This can save real and virtual storage; the amount depending on the number of resident library routines that each program uses.

If you want to use these routines but your programs are not compiled to share routines, you must recompile them and all programs must use the same level of the PL/I compiler. Programs compiled to use this facility do so automatically if the shared library is specified.

How implemented

To run PL/I application programs with the PL/I shared library, ensure that you generate the PL/I shared library modules. CICS looks for the presence of the significant shared library interface routines at startup time.

How monitored

A link-edit map shows storage savings. RMF shows overall real and virtual storage usage.

VS COBOL II

VS COBOL II programs can be loaded above the 16MB line and can also use most working storage above the 16MB line.

VS COBOL II has library modules that are grouped together in COBPACKs that need to be defined to CICS. COBPACKs can be tailored by the installation.

How implemented

One item of tailoring recommended is to have COBPACKs IGZCPPC and IGZCPAC contain only AMODE(31), RMODE(ANY) modules so that they may be loaded above the 16MB line.

VS COBOL II also has a tuning mechanism in IGZTUNE which specifies the initial amount of storage to be GETMAINed below (and above) the 16MB line.

This mechanism should be used, defining an optimum value that does not:

- Waste space below the 16MB line
- Incur unnecessary additional GETMAINs because the initial amount was too small.

How monitored

Use of IGZOPT allows VS COBOL II to report on the effect of IGZTUNE options. Production systems should ensure that such reporting is turned off.

APAR PQ16844

MJO 2/11/98

Language Environment

Language Environment (LE) conforming CICS applications issuing EXEC CICS LINK requests cause an increase in system pathlength. Repeated EXEC CICS LINK calls to the same LE-conforming program result in multiple GETMAIN/FREEMAIN requests for run-time work areas (RUWAs).

RUWAPOOL(YES) results in the creation of a run-unit work area pool during task initialization. This pool is used to allocate RUWAs required by LE-conforming programs. This reduces the number of GETMAINS and FREEMAINS in tasks which perform many EXEC CICS LINKS to LE-conforming programs.

For more information, about the RUWAPOOL system initialization parameter, see the *CICS/ESA System Definition Guide*.

Chapter 24. CICS facilities

This chapter includes the following topics:

CICS temporary storage (TS)

CICS temporary storage is a scratchpad facility that is heavily used in many systems. Data in temporary storage tends to be short-lived, emphasis being placed on ease of storage and retrieval. Temporary storage exists in two forms:

- Main temporary storage, which is in the dynamic storage area above the 16MB line (ECDSA)
- Auxiliary temporary storage is stored in VSAM-managed data set while the storage for the buffers is allocated from the ECDSA.

Temporary storage is used in many circumstances within CICS, as well as for requests from application tasks. The uses of temporary storage include:

- Basic mapping support (BMS) paging
- Message switching (CMSG transaction) or BMS routing
- Interval control: EXEC CICS START FROM (...) to hold data until it is retrieved
- Execution diagnostic facility (EDF) to review prior pages of diagnostic information
- MRO/ISC local queueing while the target system is unavailable
- Your applications for:
  - Scratchpad
  - Queueing facility
  - Data transfer.
- Other products or application packages.

Effects

If main temporary storage is used, requests to a TS queue are serialized with the storage being allocated from the ECDSA.

The performance of auxiliary temporary storage is affected by the characteristics of the data set where it resides. The VSAM control interval (CI) size affects transfer efficiency, with a smaller size being desirable if access to CIs is random, and a larger size if use of CIs is more sequential. In general, the larger the queues and write/read ratio, the more sequential the usage tends to be. Records which span control intervals are possible. Up to 32767 buffers and 255 strings can be specified, and overlap processing can be achieved, although a specific queue is still processed serially. The maximum control interval (CI) size is 64KB.

For both auxiliary and main TS queues CICS allocates a given number of entries for pointers to records in a queue (see the system initialization parameter, TSMGSET). If the number of entries is insufficient, additional storage for more pointers is acquired. This is known as TS queue extension. The number of queue extensions should normally be in the 10 to 20% range. See “Temporary storage” on page 414.

Temporary storage VSAM requests can be subtasked if SUBTSKS=1 is specified in the SIT. See “Subtasking: VSAM (SUBTSKS=1)” on page 229.

Auxiliary temporary storage queues can be made recoverable by providing a temporary storage table recovery entry. Main temporary storage can never be recoverable.

Limitations

Increasing the use of main temporary storage, using a larger CI size, increasing the number of buffers, or increasing the TSMGSET allocation increases the virtual storage needs of the ECDSA and real storage needs.

If you use auxiliary temporary storage, a smaller CI size, and a smaller TSMGSET setting if extensions are infrequent, can reduce the real storage requirements.

Recommendations

Main temporary storage

Temporary storage items are stored in the ECDSA above the 16MB line. No recovery is available. No CICS locking or enqueues are used, because there is no protection support. Queues are locked for the duration of the TS request.

The fact that temporary storage items are stored in main storage also means that there is no associated I/O, so we recommend main temporary storage for short-duration tasks with small amounts of data.

Auxiliary temporary storage

Auxiliary temporary storage occupies less address space than main temporary storage, and should be used for large amounts of temporary storage data, or for data that is to be held for long periods.

Temporary storage I/O occurs only when a record is not in the buffer, or when a new buffer is required, or if dictated by recovery requirements.

Secondary extents for temporary storage

During temporary storage initialization, the temporary storage data set is dynamically formatted. On a cold start on temporary storage (TS=(COLD,n,n), system initialization parameter) when the data set is empty, the data set is formatted to the end of the primary extent. Any secondary extents are not formatted. On a cold start of temporary storage when the data set is not empty or when temporary storage is not cold started, no further formatting of the data set takes place.

The use of secondary extents allows more efficient use of DASD space. You can define a temporary storage data set with a primary extent large enough for normal

activity, and with secondary extents for exceptional circumstances, such as unexpected peaks in activity.

It follows that you can reduce or eliminate the channel and arm contention that is likely to occur because of heavy use of temporary storage data.

Multiple buffers

The use of multiple VSAM buffers allows multiple VSAM control intervals to be available in storage at the same time. This makes it possible for the CICS temporary storage programs to service several requests concurrently, using different buffers.

If requests have to be queued, they are queued serially by temporary storage queue name. Typically, a request has to be queued if the control interval it requires is in use, or if one or more previous requests for the same queue are already waiting. Under these conditions, the servicing of requests for other queues can continue.

The use of multiple buffers also increases the likelihood that the control interval required by a particular request is already available in a buffer. This can lead to a significant reduction in the number of real input/output requests (VSAM requests) that have to be performed. (However, VSAM requests are always executed whenever their use is dictated by the requirements of physical and logical recovery.) Note that although the use of a large number of buffers may greatly improve performance for non-recoverable intrapartition destinations, the associated buffers still have to be flushed sequentially at CICS shutdown, and that might take a long time.

The number of buffers that CICS allocates for temporary storage is specified by the system initialization parameter, TS.

The benefits of multiple buffers depend on the way an installation's auxiliary temporary storage is used. In most cases, the default TS specification in the SIT (three buffers) should be sufficient. Where the usage of temporary storage is high or where the lifetime of temporary storage data items is long, it may be worthwhile to experiment with larger numbers of buffers. The buffer statistics in the CICS temporary storage statistics give sufficient information to help you determine a suitable allocation.

In general, you should aim to minimize the number of times that a task has to wait either because no space in buffers is available to hold the required data or because no string is available to accomplish the required I/O. The trade-off here is between improvement of temporary storage performance and increased storage requirements. Specifying a large number of buffers may decrease temporary storage I/O but lead to inefficient usage of real storage and increased paging.

Concurrent input/output operations (multiple strings)

Temporary storage programs issue VSAM requests whenever real input/output is required between the buffers and the VSAM temporary storage data sets. The use of multiple VSAM strings enables multiple VSAM requests to be executed concurrently, which, in turn, leads to faster servicing of the buffers.

VSAM requests are queued whenever the number of concurrent requests exceeds the number of available strings. Constraints caused by this can thus be relieved by

increasing the number of available strings, up to a maximum equal to the number of buffers.

The number of VSAM strings that CICS allocates for temporary storage is specified by system initialization parameter, TS.

Multiple strings allow more I/O operations to be performed concurrently. Several I/O requests can be outstanding at any time, up to the number of strings specified. Allowing the number of strings to default to the number of buffers ensures that no tasks are waiting for a string. Not all strings may be used in this case, however, and this causes inefficient use of storage. You should adjust the number of strings by using the peak number in use given in the statistics.

If the device containing the temporary storage data set is heavily used, the TS system initialization parameter can be used to regulate the activity, but this leads to an increase in internal CICS waits.

Control interval (CI) sizes

You should first consider whether the control interval (CI) size for the data set is suitable for your overall system requirements.

BMS paging may be on a large-screen device. Check whether it exceeds your temporary storage CI size.

Because temporary storage can use records larger than the control interval size, the size of the control intervals is not a major concern, but there is a performance overhead in using temporary storage records that are larger than the CI size.

The parameter, CONTROLINTERVALSIZE, of the VSAM CLUSTER definition is specified when you allocate your data sets.

The control interval size should be large enough to hold at least one (rounded up) temporary storage record, including 64 bytes of VSAM control information for control interval sizes less than, or equal to, 16 384, or 128 bytes of control information for larger control interval sizes. For further information about the effect of the control interval size for CICS temporary storage, see the *CICS/ESA System Definition Guide*.

How implemented

Temporary storage items can be stored either in main storage or in auxiliary storage on DASD. Main-only support can be forced by specifying TS=(,0) (zero temporary storage buffers) in the SIT.

A choice of MAIN or AUXILIARY is available for the application programmer in the WRITEQ TS command for each queue. See the *CICS/ESA Application Programming Reference* manual for programming information about the WRITEQ command.

How monitored

The CICS temporary storage statistics show TSMGSET extensions and records used in main and auxiliary temporary storage. These statistics also give buffer and string information and data on I/O activity. RMF or the VSAM catalog gives additional information on data set performance.

If recovery is used for auxiliary temporary storage, the DATA identifier (called QUEUE name by the application programmer) is enqueued for DELETEQ TS and WRITEQ TS requests but not READQ TS. In a high-activity system, DATA identifiers should be monitored to ensure that a given DATA identifier is not a resource that is constraining your transaction throughput.

You should monitor the following:

TS buffer size
This is determined by the CI size.

TS group identifier
This is controlled by the TSMGSET system initialization parameter.

TS data (QUEUE) identifiers
You should minimize their number and their duration in the system.

TS space
Make the data set allocation large enough to avoid task suspension.

Note: If the NOSPACE condition is not handled, the task is suspended until temporary storage becomes available. If the NOSPACE condition is handled (through the use of the HANDLE CONDITION NOSPACE command or the use of RESP on the WRITEQ TS command, or the WRITEQ TS NOSUSPEND command, the user receives control when the condition occurs, and can then decide whether to end the transaction normally, abend, or wait.

TS unit table (TSUT)
This is controlled by the number of different DATA identifiers created and in use concurrently in temporary storage.

Number of TS buffers
This is controlled by the second parameter of the TS system initialization parameter.

Number of TS strings
This is controlled by the third parameter of the TS system initialization parameter.

IS THIS RELEVANT TO VSE?

The 75 percent rule

Temporary storage requests on a cold-started system (i.e. with no existing auxiliary
data) are allocated from the start of DFHTEMP (the temporary storage dataset
used for storing auxiliary data). They are processed by DFHTSP, the temporary
storage program. The first control interval within DFHTEMP is used until a WRITEQ
is issued that is too long to fit into the remaining space. DFHTSP then switches to
use control interval two, etc. This process continues until 75% of the control
intervals in DFHTEMP have data written to them.

WRITEQ requests after this point are directed back to the start of the dataset. DFHTSP maintains a bytemap representing the free space available within each control interval in the dataset at any time. DFHTSP now starts interrogating the bytemap to find a control interval that can accommodate new data at, or near to, the start of DFHTEMP. The reasoning behind this is that by now, queues written earlier in the CICS run could have been deleted. Such deleted data remains in control intervals but is no longer required. If the bytemap shows a control interval contains enough space, DFHTSP reads it into a temporary storage buffer, compresses it to move all valid records to the start of the control interval, and use the remaining contiguous space to store the data from the new request.

Temporary storage attempts to reserve 25% of the control intervals in DFHTEMP to facilitate spanned record processing. This refers to data that is larger than the control intervals. Such requests generate 'special header' records used to represent the spanned data, and these records are the size of a whole control interval. As such, they require an empty control interval when being written.

If DFHTEMP contained fragmented data in each control interval, a special header record could not be stored. This is why DFHTSP tries to maintain a percentage of free control intervals for use by large items such as special header records.

CICS transient data (TD)

Transient data is used in many circumstances within CICS, including:

- Servicing requests made by user tasks, for example, a request to build a queue of data for later processing.
- Servicing requests from CICS, primarily to write messages to system queues for printing. Transient data should, therefore, be set up at your installation to capture these CICS messages.
- Managing the DASD space holding the intrapartition data.
- Initiating tasks based on queue trigger level specification and on records written to an intrapartition destination.
- Requesting logging for recovery as specified in your CICS destination control table (DCT).
- Passing extrapartition requests to the operating system access method for processing.

Various options can affect the performance of this facility.

Recovery options

Recovery can affect the length of time for which a transient data record is enqueued. You can specify one of three options:

1. **No recovery**. If you specify no recovery, there is no logging, no enqueuing for protecting resources, and no deferred work element (DWE) processing.
2. **Physical recovery**. Specify physical recovery when you need to restore the intrapartition queue to the status that it had immediately before a system failure. The main performance consideration is that there is no deferred transient data processing, which means that automatic task initiation may occur instantaneously. Records that have been written may be read by another task

immediately. CIs are released for reusable queues on the first read of a new CI. For every WRITEQ TD request, the CI buffer is written to the VSAM data set.

Note: All other resources offering recovery within CICS provide only logical recovery. Using backout in an abend situation would exclude your physically recoverable and non-recoverable transient data from the backout.

3. **Logical recovery**. Specify logical recovery when you want to restore the queues and restore trigger levels to the status that they had before execution of the failing task (when the system failed or when the task ended abnormally). Thus, logical recovery works in the same way as recovery defined for other recoverable resources such as DL/I, file control, and temporary storage.

In summary, physical recovery ensures that records are restored in the case of a system failure, while logical recovery also ensures integrity of records in the case of a task failure, and ties up the applicable transient data records for the length of a task that enqueues on them.

| Up to 32767 buffers and 255 strings can be specified for a transient data set, with serial processing only through a destination.

Specifying a higher trigger level on a destination causes a smaller number of tasks to be initiated from that destination. Transient data can participate in file subtasking if SUBTSKS=1 is specified in the SIT (see “Subtasking: VSAM (SUBTSKS=1)” on page 229).

Intrapartition transient data considerations

Multiple VSAM buffers

When you use multiple buffers and strings for intrapartition transient data support, this can remove the possible constraint in transient data caused by the use of a single system-wide buffer (and string). Statistics allow you to tune the system with regard to transient data usage.

If requests have to be queued, they are queued serially by transient data destination. Typically, a request has to be queued if the control interval it requires is in use, or if one or more previous requests for the same queue or destination are already waiting. Under these conditions, the servicing of requests for other queues or destinations can continue.

The use of multiple buffers also increases the likelihood that the control interval required by a particular request is already available in a buffer. This can lead to a significant reduction in the number of real input/output requests (VSAM requests) that have to be performed. (However, VSAM requests are always executed whenever their use is dictated by the requirements of physical and logical recovery.)

The number of buffers that CICS allocates for transient data is specified by the TD system initialization parameter. The default is three.

The provision of multiple buffers allows CICS to retain copies (or potential copies) of several VSAM CIs in storage. Several transient data requests to different queues can then be serviced concurrently using different buffers. Requests are serialized by queue name, not globally. Multiple buffers also allow the number of

VSAM requests to the transient data data set to be reduced by increasing the likelihood that the CI required is already in storage and making it less likely that a buffer must be flushed to accommodate new data. VSAM requests are still issued when required by recovery considerations.

The benefits of multiple buffers depend on the pattern and extent of usage of intrapartition transient data in an installation. For most installations, the default specification (three buffers) should be sufficient. Where the usage of transient data is extensive, it is worthwhile to experiment with larger numbers of buffers. The buffer statistics give sufficient information to help determination of a suitable allocation. In general, the aim of the tuning should be to minimize the number of times a task must wait because no buffers are available to hold the required data.

In this exercise, there is a trade-off between improving transient data performance and increased storage requirements. Specifying a large number of buffers may decrease transient data I/O and improve concurrency but lead to inefficient usage of real storage. Also, if there is a large number of buffers and a small number of queues, internal buffer searches per queue may take longer.

The buffers are obtained in the ECDSA during initialization from within the CICS region.

Multiple VSAM strings

As far as concurrent input/output operations with CICS are concerned, the transient data programs issue VSAM requests whenever real input/output is required between the buffers and the VSAM transient data data sets. The use of multiple VSAM strings enables multiple VSAM requests to be executed concurrently, which in turn leads to faster servicing of the buffers.

VSAM requests are queued whenever the number of concurrent requests exceeds the number of available strings. Constraints from this cause can thus be relieved by increasing the number of available strings, up to a maximum of 255. The limit of 255 on the number of strings should be taken into consideration when choosing the number of buffers. If the number of buffers is more than the number of strings, the potential for string waits increases.

The number of VSAM strings that CICS allocates for transient data is specified by the TD system initialization parameter. The CICS default is three.

Logical recovery

Logging, enqueuing, and DWE processing occur with logical recovery transactions, including dynamic backout of the failing task's activity on the transient data queue. Logical recovery would generally be used when a group of records have to be processed together for any reason, or when other recoverable resources are to be processed in the same task.

During processing of the transient data request, the destination queue DCT entry is enqueued from the first request, for either input or output, or both (if the queue is to be deleted), until the end of the LUW. This means that none of the other tasks can access the queue for the same purpose during that period of time, thus maintaining the integrity of the queue's status.

At the end of the LUW (syncpoint or task completion), syncpoint processing takes place and the DCT entry is logged. Any purge requests are processed (during the

LUW, a purge only marks the queue ready for purging). The empty CIs for reusable queues are released for general transient data use. Any trigger levels reached during the LUW cause automatic task initiation to take place for those queues with the TRIGLEV DCT operand specified as greater than zero. The buffer is written out to the VSAM data set as necessary.

The DEQueue on the DCT entry occurs, releasing the queue for either input or output processing by other tasks. Records written by a task can then be read by another task.

Logging activity

With **physical** recovery, the DCT entry is logged before each READQ, after the first WRITEQ for a destination, at a PURGE request, and at an activity keypoint time.

With **logical** recovery, the DCT entry is logged at syncpoint and at activity keypoint time.

Secondary extents for intrapartition transient data

During initialization of intrapartition transient data, CICS initializes a VSAM empty intrapartition data set by formatting control intervals until the first extent of the data set is filled. Additional control intervals are formatted as required if the data set has been defined with multiple extents.

The use of secondary extents allows more efficient use of DASD space. You can define an intrapartition data set with primary extents large enough for normal activity, and with secondary extents for exceptional circumstances, such as unexpected peaks in activity.

It follows that you can reduce or eliminate the channel and arm contention that is likely to occur because of heavy use of intrapartition transient data.

Extrapartition transient data considerations

Extrapartition destinations are, in practice, sequential data sets to which CICS uses QSAM PUT LOCATE or PUT MOVE commands. The main performance factor to note is the possibility of operating system waits; that is, the complete CICS region waits for the I/O completion. The wait (of long duration) can occur for one of the following reasons:

- No buffer space available.
- Secondary space allocation.
- Volume (extent) switching.
- Dynamic open or close of the data set.
- A force end of volume caused by the application.
- The data set is defined on a physical printer (1403 or 3211) and the printer has run out of paper.
- A RESERVE has been issued for another data set on the same volume.

Therefore, you should try to eliminate or minimize the occurrences of CICS region waits by:

- Having sufficient buffering and blocking of the output data set
- Avoiding volume switching by initially allocating sufficient space

- Avoiding dynamic OPEN/CLOSE during peak periods.

An alternative method of implementing sequential data sets is to employ a CICS user journal. Table 12 summarizes the differences between these two methods.

Table 12. Extrapartition transient data versus user journal

Extrapartition TD	**User Journal**
Region (CICS) may wait	Task waits
Buffer location: In MVS storage	Buffer location: In DSA
Number of buffers: 1—32767	Two buffers
Input ***or*** output	***Both*** input ***and*** output, but tasks may wait
Accessible by multiple tasks	• Accessible for output by multiple tasks • Accessible for input by single task under exclusive control

Indirect destinations

To avoid specifying extrapartition data sets for the CICS-required entries (such as CSMT and CSSL) in the DCT, you are recommended to use indirect destinations for combining the output of several destinations to a single destination. This saves storage space and internal management overheads.

Long indirect chains can, however, cause significant paging to occur.

Limitations

Application requirements may dictate a lower trigger level, or physical or logical recovery, but these facilities increase processor requirements. Real and virtual storage requirements may be increased, particularly if several buffers are specified.

How implemented

Transient data performance is affected by the REUSE, TRIGLEV, and DESTRCV operands in the TYPE=INTRA macro in the destination control table (DCT).

Recommendations

Suggestions for reducing WAITS during QSAM processing are to:

- Avoid specifying a physical printer.
- Use single extent data sets whenever possible to eliminate WAITS resulting from the end of extent processing.
- Avoid placing data sets on volumes on volumes subject to frequent or long duration RESERVE activity.
- Avoid placing many heavily-used data sets on the same volume.
- Choose BUFNO and BLKSIZE such that the rate at which CICS writes or reads data is less than the rate at which data can be transferred to or from the volume, for example, avoid BUFNO=1 for unblocked records whenever possible.
- Choose an efficient BLKSIZE for the device employed such that at least 3 blocks can be accommodated on each track.

How monitored

The CICS statistics show transient data performance. CICS transient data statistics can be used to determine the number of records written or read. Application knowledge is required to determine the way in which the lengths of variable length records are distributed. RMF or the VSAM catalog shows data set performance.

CICS monitoring facility

The CICS monitoring facility collects data about the performance of all user- and CICS-supplied transactions during online processing for later offline analysis. The records produced by CICS monitoring are of the MVS System Management type 110, and are written to an SMF data set.

Monitoring data is useful for performance, tuning, and for charging your users for the resources they use. See Chapter 6, “The CICS monitoring facility” on page 65 for further information.

Limitations

Performance class monitoring can be a significant overhead. The overhead is likely to be about 5 to 10%, but is dependent on the workload.

Recommendations

If you do not need accounting information because other billing processes exist and you have other means of gathering any performance data required, the CICS monitoring facility should not be used. The same is true for the exception component.

Recording of the above information incurs overhead, but, to tune a system, both performance and exception information may be required. If this is not a daily process, the CICS monitoring facility may not need to be run all the time. When tuning, it is necessary to run the CICS monitoring facility during peak volume times because this is when performance problems occur.

Consider excluding fields from monitoring records if overuse of the SMF data set is a potential problem.

How implemented

To implement CICS monitoring, you can reset the system initialization table parameters (MNPER, MNEVE, MNEXC, and MN) see the *CICS/ESA System Definition Guide*.

You can change the settings dynamically using either CEMT INQUIRE|SET MONITOR or EXEC CICS INQUIRE|SET MONITOR. See “Controlling CICS monitoring” on page 72 for more information. Alternatively see the *CICS/ESA CICS-Supplied Transactions* manual for details of CEMT; see the *CICS/ESA System Programming Reference* for programming information about INQUIRE and SET commands.

For further information on using SYSEVENT with the CICS monitoring facility and with RMF, see Chapter 6, “The CICS monitoring facility” on page 65.

How monitored

CICS Monitoring Domain statistics show the number of records produced of each type. These statistics monitor CMF activity.

RMF statistics gathered with and without monitoring active give an indication of the overhead incurred.

CICS trace

CICS trace is used to record requests made by application programs to CICS for various services. Because this involves the recording of these requests each time they occur, the overhead depends on the frequency of the requests.

The CICS internal trace table resides in MVS virtual storage above the 16MB line (but not in the EDSAs).

A trace table always exists and is used for recording exception conditions useful for any first failure data capture. Other levels of trace are under the control of the user. There are a large number of parameters and the CEMT commands which allow dynamic control over the system and transaction dumps.

Effects

Buffers for the CICS auxiliary trace data set are allocated dynamically from MVS free storage below the 16MB line. Auxiliary trace is activated when the system initialization parameter AUXTR, or a startup override, is set on.

Buffer allocation may also take place at execution time in response to a CETR or CEMT transaction request to set auxiliary trace to START (CEMT SET AUXTRACE START) or simply to open the auxiliary trace data set. See the *CICS/ESA CICS-Supplied Transactions* manual for further information.

Deallocation or freeing of the buffer space occurs in response to CEMT SET AUXTRACE STOP command. Note that the buffer space is ***not*** freed on STOP and SWITCH requests, the former not implying CLOSE and the latter having been optimized.

Limitations

Running trace increases processing requirements considerably. Not running trace, however, reduces the amount of problem determination information that is available.

The additional cost of auxiliary trace is very largely due to the I/O operations. Auxiliary trace entries vary in size, and they are written out in blocks of 4KB. Twin buffers are used but, even if the I/O can be overlapped, the I/O rate is quite large for a busy system.

When you use CICS auxiliary trace, you may need to decrease the relevant DSASZE system initialization parameter by 8KB to ensure that adequate address space is given up to the operating system to allow for the allocation of the two 4KB auxiliary trace buffers.

Recommendations

The trace table should be large enough to contain the entries needed for debugging purposes.

With first failure data capture, CICS produces some trace entries regardless of the settings produced. Because of this most of the tracing overhead can be reduced by running with the following options:

- Internal tracing off
- Auxiliary tracing on
- Print auxiliary trace data only when required.

CICS allows tracing on a transaction basis rather than a system basis, so the trace table requirements can be reduced.

How implemented

Trace activation is specified with the INTTR system initialization parameter or as a startup override.

The size of the trace table is specified by the TRTABSZ system initialization parameter or as a startup override. The minimum size is 16KB.

Trace can be defined at the transaction level with the TRACE keyword on in the TRANSACTION definition.

Auxiliary trace activation is specified with the AUXTR system initialization parameter.

With CICS initialized and running, internal trace and auxiliary trace can be turned on or off, independently and in either order, with one of the following: CETR, CEMT SET INTRACE START or CEMT SET AUXTRACE START commands. Auxiliary trace entries are recorded only when internal trace is active.

How monitored

No direct measurement of trace is given. RMF can show processing and storage requirements.

CICS recovery

Some types of recoverable resources, when they are accessed for update, cause logging. Do not define more resources as recoverable than you need for application programming requirements, because the extra logging incurs extra I/O and processor overheads. If the resource in question does not require recovery, these overheads are unproductive.

Limitations

Specifying recovery increases processor time, real and virtual storage, and I/O requirements. It also increases task waits arising from enqueues on recoverable resources and system log I/O, and increases restart time.

Recommendation

Do not specify recovery if you do not need it. If the overhead is acceptable, logging can be useful for auditing, or if a data set has to be rebuilt.

For information on specific recoverable resources, see “VSAM string settings for LSR (STRNO)” on page 226, “CICS journaling (BUFSIZE, SYSWAIT=STARTIO|ASIS)” on page 250, and “Maximum keylength for LSR (KEYLEN)” on page 227.

How implemented

See the *CICS/ESA Recovery and Restart Guide* for information on each resource to be specified as recoverable.

How monitored

CICS auxiliary trace shows task wait time due to enqueues. RMF shows overall processor usage. CICS monitoring data shows task wait time due to journaling.

CICS security

CICS provides an interface for an external security manager (ESM), such as RACF, for three types of security: transaction, resource, and command security.

Effects

Transaction security verifies an operator’s authorization to access a transaction. Resource security limits access to data sets, transactions, transient data destinations, programs, temporary storage records, and journals. Command security is used to limit access to specific commands and applies to special system programming commands. For example, EXEC CICS INQUIRE, SET, PERFORM, DISCARD, and COLLECT. Transactions that are defined with CMDSEC=YES must have an associated user.

Limitations

Protecting transactions, resources, or commands unnecessarily both increases processor cycles, and real and virtual storage requirements.

Recommendations

Because transaction security is enforced by CICS, it is suggested that the use of both resource security and command security should be kept to the minimum. The assumption is that, if operators have access to a particular transaction they therefore have access to the appropriate resources.

Unless it is vital, do not use the PERFORM SECURITY REBUILD command. It severely slows down your system. This is, because, while it is being processed, all transactions doing a security check must wait until it has completed.

How implemented

Resource security is defined with:

- The RESSEC(YES) keyword in the TRANSACTION definition.

Command security is defined with:

- The CMDSSEC(YES) keyword in the TRANSACTION definition.

How monitored

No direct measurement of the overhead of CICS security is given. RMF shows overall processor usage.

CICS storage protection facilities

There are three facilities available that are related to storage protection:

- Storage protect
- Transaction isolation
- Command protection.

Each offers protection as follows:

Storage protect

Protects CICS code and control blocks from being accidentally overwritten by user applications.

Transaction isolation

Offers protection against transaction data being accidentally overwritten by other user transactions.

Command protection

Ensures that an application program does not pass storage to CICS using the EXEC CICS interface, which requires updating by CICS, although the application itself cannot update the storage.

Recommendation

Storage protection, transaction isolation, and command protection protect storage from user application code. They add no benefit to a region where no user code is executed, a pure TOR or a pure FOR (where no DPL requests are function shipped).

Transaction isolation and applications

When using transaction isolation it is necessary to “activate” pages of storage to the task's allocated subspace. Before the storage is activated to the subspace it is fetch protected and, so, the task cannot access the storage. After it is activated to the subspace allocated to the task, the task has read/write access to the storage. CICS needs to activate user storage to a subspace every time the user task getmains a new page of user key task lifetime storage. Some performance cost is involved when activating storage to a subspace, so the activity should be kept to a minimum.

Storage below the 16MB line is activated in multiples of 4KB. Storage above the line is activated in multiples of 1MB. A user task rarely requires more than 1MB of storage. So a user task that executes completely above the line only requires one activate.

It is recommended that all programs should be link edited using RMODE(ANY) and defined DATALOCATION(ANY). All transactions should be defined TASKDATALOGIC(ANY), thus reducing the number of storage activations. Where it is necessary to obtain storage below the line, performance can be improved by obtaining all the storage in one getmain rather than several smaller getmains. This also keeps the number of storage activates to the minimum.

Certain programming languages, such as PL/I and VS COBOL, obtain storage below the line. COBOL II offers better performance because it obtains all its storage above the line.

Chapter 25. Tuning XRF

This chapter includes the following topics:

Using extended recovery facility (XRF)

The extended recovery facility (XRF) is a related set of programs that allows an installation to achieve a higher level of availability to end users.

Availability of a system is a function of the number of outages and the duration of each outage. Even if all unplanned outages caused by hardware and software failures could be eliminated, there would still be planned outages for maintenance, configuration changes, or migration.

CICS with XRF aims to reduce the duration of unplanned outages by having an **alternate system** that monitors the well-being of the **active system** that is running the CICS workload. (The alternate and active systems can be on the same or separate MVS images.) The alternate system reacts automatically to a failure of the active system, and then starts to **take over** from the failing system. Because a takeover is faster than the restart of a CICS system, planned outages are also speeded up by having the alternate system take over from the active system.

Note that XRF itself does not eliminate outages. Because XRF reduces the effects of planned and unplanned outages on the end user, it is able to provide a higher level of availability than a non-XRF CICS system, but it does not provide continuous availability.

The main source of information on CICS with XRF is the *CICS/ESA 3.3 XRF Guide*. This section provides a summary of performance and tuning considerations for XRF.

Recovery time

The length of outage perceived by the end user in an XRF environment depends on many factors, most of which are common to the non-XRF environment as well. A takeover has much in common with an emergency restart, so factors such as the number of backouts and the amount of log to be read back are as important to takeover times as emergency restart times. The main differences in a takeover are due to the way that a failure is detected, the way that terminal sessions are handled during the takeover, and the semi-initialized state of the alternate system.

Failure detection

The alternate CICS continuously monitors the 'health' of the active CICS system and, if it detects that it has failed, it initiates a takeover. In a typical non-XRF environment an operator detects the failure of the CICS system and starts procedures to cancel the system (if necessary) and then starts a new system. Many users find that this automation of failure detection and recovery initiation causes the biggest reduction in outage time.

Session switching

In an XRF system, Class 1 terminals have a backup session bound for them by the alternate system. Class 1 (principally remote VTAM SNA) terminals are also called **XRF-capable** terminals. At takeover, these terminals can be switched to the backup session.

Non XRF-capable sessions are divided into two classes. Class 2 sessions are rebound automatically at takeover and Class 3 sessions require operator intervention before they can be used again.

VTAM defines whether a terminal can be Class 1. For details of which terminals are XRF-capable, see the *CICS/ESA 3.3 XRF Guide* and the relevant VTAM manuals.

The time taken to restore service to the end user in an XRF system depends on the class of terminal being used:

- **XRF-capable session (Class 1)**: The major difference between an XRF takeover and an emergency restart is in the way that terminal sessions are re-established after a failure occurs. In a non-XRF environment, all the sessions are broken and must be re-established after emergency restart has completed. This can be a time-consuming process, requiring relatively large amounts of processor time.

  With XRF, however, Class 1 terminal sessions are switched to their backup sessions. This is mainly done by the boundary network node network control program (NCP), and can be done in parallel with other restart activities. The recovery of these sessions, which includes 'clean-up' and sending of recovery messages, is in the critical path of takeover, but is a very quick process. Therefore, the greater the number of Class 1 sessions attached to the CICS system, the greater the benefit of XRF when a failure occurs in terms of reduced outage time.

  After XRF is fully implemented, the size of the network becomes a less important factor in outage time. Session clean-up time does increase as the number of terminals increases, but the clean-up time is a very small proportion of the overall takeover time.

  The time for the alternate CICS system to complete emergency restart and backout may be the primary factor in outage time for Class 1 terminal users. Outage time can also be extended by contention for real storage between the alternate system and other workloads.

- **Non XRF-capable sessions (Class 2)**: For Class 2 sessions, the network size has the same effect as in a non-XRF environment. All the terminals that are in session at takeover time have to be rebound, which is a time-consuming process. The users of these terminals gain from the automatic detection of failure, however.
- **Non XRF-capable sessions (Class 3)**: Class 3 terminals have to wait for the operator to switch them to the new active, and this is probably the major factor in the time taken to restore service to them.

Semi-initialized alternate system

A takeover by an alternate system is faster than restarting a CICS system because, in addition to the factors pointed out earlier in this section, the alternate system is partially initialized and has already completed some of the necessary steps, including data set allocation and loading of the nucleus.

Performance of XRF phases

Takeover is only one of a series of phases that an XRF system goes through. The phases are initialization, synchronization, surveillance and tracking, takeover, and termination. The functional aspects of these phases are described in the *CICS/ESA 3.3 XRF Guide*. In this section we examine the performance aspects of these phases.

Initialization and termination phases

These phases are almost identical to their non-XRF counterparts and are not discussed at further length here.

Synchronization phase

In the synchronization phase, the active sends an image of the trackable resources (TCT and session states) to the alternate system. With this information the alternate system builds a TCT. It then opens backup sessions for the XRF-capable terminals.

A system transaction (CXCU) runs on the active system to send the image of the TCT and sessions to the alternate system. This transaction has a measurable effect on processor utilization and throughput, which is greater on a small processor. The effects can be mitigated to some extent by changing the priority of CXCU; for details, see the *CICS/ESA 3.3 XRF Guide*.

For the alternate system, the processor cost of synchronization is in two parts.

1. The cost of building the TCT, which can be relatively significant on a small processor
2. The cost of binding the standby sessions which is principally a VTAM processor cost and can be considerable on any size of processor, depending on network size.

These costs affect the active system if the alternate system is on the same MVS image.

The real storage demands of the active and alternate systems also increase during synchronization, the latter particularly.

Because this phase is relatively short (a few minutes) and normally happens only once before takeover, you may consider scheduling the synchronization phase to a period of lower activity, when its effect is less. A catchup for a system with few XRF-capable sessions, and especially one with few or no terminals (such as a FOR), has limited effect. The full benefits of XRF are available only when an alternate system has synchronized with the active system.

Surveillance and tracking phase

The active/alternate pair spends most of its time in the surveillance and tracking phase. The alternate system monitors the active system to detect any failure. Changes to the TCT and session states that occur on the active system are reflected on the alternate system.

The cost of surveillance is a small, constant processor overhead and an increase in real storage working set size. This cost is present on an active system that has XRF=YES coded, whether or not an alternate system is running. The cost of running an alternate system that is carrying out surveillance is comparable with the overhead of having XRF=YES on an active system. External throughput and response time on the active system are not significantly affected.

The cost of tracking is determined by the rate of change to the TCT (CEDA and autoinstall) and session states (autoinstall and logon/logoff). On an active system, the extra cost of passing information to the alternate system is not much more than the cost of surveillance. For the alternate system, the cost of tracking and the cost of installation and binding sessions are almost the same as the cost of these processes on the active system.

Note: For a system autoinstalling XRF-capable sessions which is being tracked by an alternate system, the cost of autoinstall over the whole system is double the cost on a non-XRF system. This is due to the ‘mirroring’ of the actions of the active system by the alternate system.

Takeover phase

The *CICS/ESA 3.3 XRF Guide* discusses takeover.

Active CICS, VTAM session close: When the active CICS ends, its VTAM APPL is closed. The VTAM processing (closing all the active sessions) can delay the termination of the CICS address space. The alternate CICS system waits until the active system has terminated before proceeding, and so takeover time can be lengthened.

For higher availability, a command can be placed in the alternate system’s CLT which, when used in conjunction with a correctly configured MVS IEAIPS, causes the active system to be moved to a higher dispatching priority than VTAM while it dumps and abends. This technique is of most use on a uniprocessor.

Clock synchronization: In a two-MVS image, when the active CICS system has terminated, the alternate CICS system can continue to take over its resources, but only if the time-of-day clock reading on the alternate system’s MVS image is later than that on the active system’s MVS image at termination. If it is not, the alternate system waits until it is. This is necessary because CICS often relies on time-of-day (TOD) clock values to provide the correct sequence of events; for example, when system log records are used for backout or recovery.

This means that, for best takeover time, the alternate system's clock time must be equal to or ahead of the active system's time. Ideally, the clocks should be exactly synchronized. If the alternate system is ahead of the active system at one takeover, on a reverse takeover it is behind and has to wait.

In practice, however, exact synchronization is impossible to achieve because individual TOD clocks drift at different rates. As long as the clocks are within a few seconds of each other, takeovers is not seriously delayed.

Emergency restart: The emergency restart-like part of a takeover has many of the same considerations as a normal emergency restart. In an XRF environment, some considerations are given added emphasis:

- Effect of data backout. Because XRF is a means of automating the detection of a failure and the subsequent emergency restart, it follows that data backout with XRF during the takeover process is the same as it is in any other emergency restart. The backout process is the same, and the time it requires (as with any emergency restart) is a function of the uncommitted updates to protected resources of in-flight tasks at the time of failure, completely independent of XRF. However, these updates do have to be backed out as part of the takeover process.

  Data backout is usually a greater portion of overall takeover time on a larger processor, because:

  – Although the same random behavior of in-flight transactions exists for any processor, a larger processor running at the same processor utilization is likely to have more transactions in flight at the time of failure. Therefore, the potential for, and the probability of, data backout increases with the size of the processor at a given utilization.

  – A faster processor completes the processor-bound portions of takeover faster than a smaller processor with the same terminal network size. These activities are primarily VTAM activities, such as session cleanup on the alternate system and closing the sessions on the active system. Because the faster processor tends to make the overall takeover time less, the I/O-bound activities such as data backout becomes a more significant portion of the total.

  In order to reduce the number of backouts, it is important to design long-running transactions so that they reach syncpoint as soon as possible when recoverable resources can be committed.

- Activity keypointing. It is a good idea to have a low AKPFREQ value because this may reduce the LOG read time. CICS needs to read back at least as far as the last activity keypoint to ensure that it knows about all the tasks to be backed out. For details, see "Activity keypoint frequency (AKPFREQ)" on page 249.

- Other jobs on the MVS image. Other activity on the MVS image can affect emergency restart time by reducing the resources available to CICS. It is wise to consider real-storage isolation (fencing).

  For details of this effect see "Isolating (fencing) real storage for CICS (PWSS and PPGRTR)" on page 183.

LU clean-up and recovery: XRF-capable sessions are switched early on in takeover; the switching takes place in parallel with other takeover activities. Toward the end of takeover, these sessions are cleaned up and recovered. This process involves sending VTAM commands and user messages (if specified) to the sessions, the rate of arrival of which is determined by the network layout. CICS can recover terminals very quickly in comparison with binding sessions for them.

Class 2 and Class 3 terminals have to be rebound after takeover. This is a time-consuming process and is also resource intensive. The AUTCONN parameter in the SIT gives a means of delaying the re-bind of Class 2 terminals, so as to allow the XRF-capable session users to get more work done earlier. For highest availability, all sessions should be XRF-capable.

Post-takeover stabilization: After takeover completes, there is a period in which the new active system adjusts to running the CICS workload. This period is similar to just after an emergency restart, except that users of XRF-capable sessions can enter transactions earlier, and so the build-up of transaction rate may be faster.

The use of FILSTAT=(CLOSED,ENABLED) in the FCT may help to reduce the effect of file opening on the system, at the expense of response time for the transaction that accesses the file. However, in a system where all the files are needed almost immediately, this has little overall effect.

The definition of the alternate system to MVS must ensure that the CICS workload can be supported after takeover. Commands can be imbedded in the CLT to change the alternate system's dispatching priority or change the priority of other work at takeover time.

Cost of XRF

Processor utilization

In an environment where CICS is running and the processor workload is low enough to give satisfactory throughput and response times, the addition of XRF should not place a significant additional demand on the processor. This statement applies to systems other than those which have a large autoinstall or logon/logoff rate. In such cases, because the alternate system duplicates the actions of the active system, the total demand of the active/alternate pair for processor resource for these activities is double that of a non-XRF system.

Processor utilization is most likely to be a factor for short periods when specific events occur:

- When the alternate system signs on to the CICS availability manager (CAVM), the active system is notified and starts sending over session information and status to it. The alternate system starts acquiring backup sessions for Class 1 terminals. This can cause a large increase in processor utilization on the alternate system and a smaller increase on the active system for this catchup phase.
- At takeover time, the sessions have to be ended on the active MVS image. VTAM activity causes a noticeable effect on processor utilization during this time.

- At takeover time there is a noticeable increase in processor utilization while VTAM goes through session cleanup for sessions that were established at the time of failure.

  This becomes a bigger factor if the MVS image on which the alternate system is running is also the network owner in a dual-MVS image environment. The VTAM network owner has to terminate the old sessions. This subject is discussed in the *CICS/ESA 3.3 XRF Guide*.

Real storage

Using the XRF function of CICS increases the real storage demands of your system. This cost is in two &partops.:

1. **The active system**: Using XRF=YES on an active system increases the real storage requirements of that CICS address space mainly because of the CAVM code and associated data.
2. **The alternate system**: Each alternate CICS is a separate address space which has its own CAVM data. You can economize on the real storage to some extent by placing the CAVM code in the ELPA if you have more than one CICS system with XRF=YES running on the same copy of MVS. The real storage requirement during the surveillance phase, if there is no tracking, is much lower than that of an active CICS system; this is typical of alternate AORs and FORs because there is hardly any tracking for them to do. When the system is in the synchronization phase, or when session state and TCT changes are being tracked by the alternate system, the real storage requirements increase to be closer to those of an active CICS system at a low transaction rate.

Virtual storage

Most of the new code for XRF and its control blocks is located above the 16MB line. Some additional storage is used below the line for:

- VSAM control blocks for the CAVM data sets
- Storage for an extra CICS system task
- MVS storage for the XRF subtask.

Tuning XRF performance

Control and message data sets

The active and alternate pair of CICS systems communicates using a pair of shared data sets. These data sets are called the control and message data sets. The flow of data to and from these data sets is managed by the CICS availability manager (CAVM). The CAVM exists on both the alternate and the active CICS system. Each CAVM writes a surveillance record every two seconds and reads the data set to determine the “health” of the other CICS. In addition, when the alternate CICS system is initialized, the active CICS system sends messages to the alternate system (via the CAVM) about its TCT and session states using the message data set.

These data sets are important for each XRF pair, and they should be placed on DASD with reasonable response time. In addition, the control and message data sets for any given pair of active and alternate systems should be placed on separate actuators, controllers, and channel paths. If I/O to the control data set becomes impossible, the CAVMs attempts to exchange status information by means of the message data set.

The method of space calculation, more precise recommendations on data set placement, and the appropriate definition parameters and JCL for the CAVM data sets are given in the *CICS/ESA System Definition Guide*.

Alternate CICS system initialization

Unless your system contains Class 3 terminals, specify TCT=NO in the SIT for the alternate system. If you do not, takeover can be delayed due to merging Class 1 and Class 2 terminal definitions.

Clock synchronization between active and alternate CICS systems

As mentioned under “Takeover phase” on page 314, to ensure optimum takeover times, you should try to keep the system clocks as nearly synchronized as possible.

Backout processing time

The time for the recovery utility program (DFHRUP) to complete backout processing is important both to fast emergency restarts and to fast takeovers. One factor in this time is how far back the utility must read on the system log before it finds a keypoint record. Read the discussions of AKPFREQ values in “Activity keypoint frequency (AKPFREQ)” on page 249 to see how this log read time can be minimized.

Another factor that affects this time is the number of backouts to be done, which is a function of the application design of the inflight tasks. Frequent syncpoints should be designed into long running tasks that update protected resources.

VTAM virtual storage

You may want to limit VTAM’s use of virtual storage during takeover.

VTAM’s I/O buffers reside in the MVS CSA below the 16MB line. During an XRF takeover, the amount of communication between the host and the NCP(s) increases significantly. This is due to the SWDT messages going to the NCPs (one message per Class 1 terminal), and the NCP responses to these messages. This results in greater use of VTAM’s I/O buffers during takeover than during steady state processing. In this situation, VTAM can expand its I/O buffer pool to accommodate this additional demand by acquiring additional CSA.

This expansion depends heavily on the IOBUF size specified, the point when expansion occurs, and the number of Class 1 terminals in the network.

Some installations with virtual storage constraints may not be able to allow VTAM to expand its I/O buffers a great deal. Expansion of these buffers is controlled by the IOBUF parameter in the ATCSTRxx member of SYS1.VTAMLST.

MVS/ESA IEAIPS parameters

Storage isolation for the alternate CICS system

Takeover time is very sensitive to other work on the same MVS image as the alternate system, especially if it results in excessive paging and real storage contention with the alternate CICS system. It is important to ensure that the minimum working set size for the alternate CICS system is reserved, by storage isolation, in order to minimize takeover time and to ensure reliable operation of the alternate system. This can be accomplished using the PWSS and PPGRTR parameters in the IEAIPSxx parmlib member of MVS/ESA. For further information, see “Isolating (fencing) real storage for CICS (PWSS and PPGRTR)” on page 183.

Dispatching priority for the alternate CICS system

It is essential to the proper functioning of the surveillance mechanism that an alternate CICS system has a dispatching priority, relative to other address spaces, which means it is dispatched when it needs to be, and does not have to wait for a lull in other work. Otherwise the alternate system may take longer to detect a failure of the active system. Also, the active CICS system may conclude that the alternate system has failed because its surveillance signals are absent.

Dispatching priority for failing active system

The completion of the SDUMP and the ABEND processing of the failing active system is important to the overall takeover time, because the active system must terminate before the alternate system can complete its own initialization.

To minimize the time for this activity, the dispatching priority of CICS, which is normally less than VTAM’s, can be set above VTAM’s in the MVS image with the failing active system. To enable this, a new domain is defined with a higher dispatching priority than that of VTAM, yet lower than that of JES, to accommodate the temporary change initiated by a command in the CLT during takeover processing. Although VTAM must close the LU sessions with the failing active system, this activity is not in the critical path of takeover processing.

This technique is of particular benefit in a dual-MVS image environment where the active system is running on a small uniprocessor.

For the specific definition and format of the domain definitions and specification of dispatching priorities in IEAIPSxx, see the *MVS/ESA Initialization and Tuning Guide*.

Alternate delay interval (ADI)

The ADI=(**30**|*decimal-value*) system initialization parameter is used on an XRF system to define the alternate delay interval, in seconds. This is the interval that must elapse between the (apparent) loss of surveillance signal from the active system and any reaction by the alternate system.

An installation can use a low ADI and TAKEOVER=MANUAL, in which case the operator can make the decision as to whether a takeover is really required.

You can set surveillance off with the CEBT command before undertaking a procedure that would otherwise result in an unwanted takeover, such as taking an

MVS dump of the active system. After the procedure has completed, you can set surveillance on again with the CEBT command.

Limitations

Low ADI provides early warning of the failure of the active system but causes no extra overhead. If set too short, it increases the likelihood of an unwanted takeover. This is particularly true if TAKEOVER=AUTO is being used. It should be set long enough so that expected events in the system are not detected as failures of the active system. If events such as system dumps that cause the active system not to be dispatched for 30 seconds occur regularly, for example, ADI should be set to a value of more than 30. If this value is set too high, it increases takeover time and, therefore, decrease availability.

Recommendation

Set ADI to a large enough value so that common events which cause the active system to be delayed are not detected as failures, resulting in an unwanted takeover.

JES delay interval (JESDI)

The JESDI=(**30**|*decimal-value*) system initialization parameter is used to define the JES delay interval, in seconds. The alternate system waits for this period of time before involving the system operator if the active system does not terminate.

Limitations

In a two-MVS image configuration, a small JESDI can provide early warning of MVS image failures. If it is set small enough, the message asking the operator to confirm job termination or MVS image failure always appears. This does not hinder smooth operation of the XRF function, but installations may not wish the message to appear except under unusual circumstances. One reason is that an incorrect reply to this message can destroy the integrity of the system's data.

Recommendation

The main factor affecting the choice of JESDI is the time taken for the SDUMP of the active system (if any). To avoid the operator message appearing except in cases where operator intervention is probably needed, JESDI must be set larger than the expected SDUMP time.

Primary delay interval (PDI)

The PDI=(**30**|*decimal-value*) system initialization parameter is used to define the primary delay interval, in seconds. This is the interval that must elapse between the (apparent) loss of surveillance signal from the alternate system and any reaction by the active system.

Recommendation

Set PDI to a large enough value so that common events which cause the alternate system to be delayed are not detected as possible failures.

Recovery options (RECOVOPTION and RECOVNOTIFY)

The TCT options for XRF determine what action is taken for Class 1 terminals after sessions are switched during takeover. The **RECOVOPTION** option allows you to influence the way in which CICS recovers the terminal's session, and returns the terminal to service after takeover. The **RECOVNOTIFY** option enables you to specify the notification to be sent to the user of a Class 1 terminal after a takeover.

The RECOVOPTION=(**SYSDEFAULT**|NONE|CLEARCONV|RELEASESESS|UNCONDREL) TCT parameter provides control over the manner in which the terminal is recovered after takeover. The default value (SYSDEFAULT) allows CICS to determine the optimum procedures to recover the terminal. The other values force CICS to adopt more restrictive procedures.

The RECOVNOTIFY=(**NONE**|MESSAGE|TRANSACTION) parameter allows you to choose whether to send a recovery message to the terminal at takeover. Your decision depends on whether you wish the takeover to appear transparent to some terminal users.

For further information on these parameters, see the *CICS/ESA 3.3 XRF Guide*.

Limitations

The options on RECOVNOTIFY can have an effect on overall takeover times, as follows:

- The default is **NONE** (no notification), and this choice results in the fastest takeover because no recovery messages are sent to the terminals.
- The **MESSAGE** option causes a recovery message to be sent. This message can be either a BMS map supplied by CICS, or an alternate map name that you specify. Because this option requires BMS to format and send out maps, there is some increase in session recovery times, which depends on the complexity of the map to be sent. More importantly, the recovery time perceived by the end-user is affected by the time it take to send the messages through the network.
- The third option, **TRANSACTION**, can slow down takeover significantly. It causes the specified transaction to be initiated and attached to the terminal. Its effect depends on exactly what the specified transaction does, but always involves the overhead of task initiation, which includes storage and processor resources. TRANSACTION should be used with care.

Recommendation

There may be cases where it is necessary or desirable to ensure that the end user is made aware that a takeover has taken place. This can be done by sending a recovery message, or by initiating a transaction to the terminal. Because these are specified by terminal, the options can be different for different terminals.

Initial data set status (OPENTIME or FILSTAT)

Setting the RDO OPENTIME FILE operand or the FILSTAT operand in the FCT can affect the speed at which the system recovers after takeover.

If OPENTIME(FIRSTREF) or FILSTAT=(ENABLED,CLOSED) is coded, it leaves the data sets closed until the first transaction references them, unless they are needed for backout of inflight tasks. The first transaction to refer to a data set, and any others that access it while it is being opened, suffers increased response time, however. In a system with a large number of data sets that are referenced infrequently, this option spreads the effect of data set opening at the cost of increased individual transaction time.

If OPENTIME(STARTUP) or FILSTAT=(ENABLED,OPENED) is coded instead, all data sets are opened immediately after takeover. On systems where all the data sets are referenced soon after takeover, transaction response time is affected while the data sets are being opened initially, but the period for the average transaction response time to recover to the level prior to takeover may be shortened.

AUTCONN

AUTCONN=(0|*hhmmss*) is used to define an additional delay between the completion of takeover and the first attempt to re-bind Class 2 sessions.

CICS automatically gives a delay of one second per ten Class 1 terminals that were in session at the time of takeover, up to a maximum of four minutes. By setting this delay, users of Class 1 terminals are given a period in which they can process more work at the expense of users of Class 2 sessions.

Note: This additional delay also applies to terminals with CONNECT=AUTO specified in the TCT on cold, warm, and emergency restarts.

Chapter 26. Improving CICS startup and normal shutdown time

This chapter tells you how to try to reduce the amount of time for CICS startup. Because various configurations are possible with CICS, different areas of the startup may require attention. Shutdown is discussed in the section on Buffer Considerations.

1. Start by defining your GCD, LCD, CSD and RSD, as shown in the *CICS/ESA System Definition Guide*.
2. When defining your terminals, pay attention to the position of group names within the GRPLIST. If the group containing the TYPETERMs is last, all the storage used for building the terminal definitions is held until the TYPETERMs are known and this could cause your system to go short-on-storage.

   Groups in the GRPLIST in the SIT are processed sequentially. Place the groups containing the model TERMINAL definitions followed by their TYPETERMs in the GRPLIST before the user transactions and programs. This minimizes the virtual storage tied up while processing the installation of the terminals.

   Note: All terminals are installed, even surrogate TCT entries for MRO.

   You must ensure that the DFHVTAM group precedes any TERMINAL or TYPETERM definition in your GRPLIST. It is contained in the DFHLIST GRPLIST, so adding DFHLIST first to your GRPLIST ensures this. If you do not do this, the programs used to build the TCT is loaded for each terminal, thus slowing cold start.
3. You should not have more than about 100 entries in any group defined in the CSD. This may cause unnecessary overhead during processing, as well as making maintenance of the group more difficult.
4. Make sure that changing the START= parameter does not change the default for any facilities that your users do not want to have AUTO-started. Any facility that you may want to override may be specifically coded in the PARM= on the EXEC statement, or all of them may be overridden by specifying START=(...,ALL).
5. Tune the VSAM parameters of the local and global catalogs to suit your installation.
   a. CI sizes should be changed for optimum data and DASD sizes (see “Size of control intervals” on page 217 for more information). 4KB index CI, and 8KB or 16KB data CI can be recommended; 32KB data has been found to slow down the COLD start.
   b. We recommend that you specify the BUFNI and BUFND parameters in your JCL for the GCD via the AMP= parameter, rather than using BUFSPACE.
   c. Allocate not more than ten data buffers. CICS opens the GCD with 32 strings, so allocate one per string, plus one more.

d. Alter the number of index buffers by coding the number of strings plus the number of index set records in the index. The number of records in the index set can be calculated from IDCAMS LISTCAT information as follows:

- T = total number of index records (index REC-TOTAL)
- D = data control interval size (data CISIZE)
- C = data control intervals per control area (data CI/CA)
- H = data high-used relative byte address (data HURBA)
- The number of index set records can then be computed:

  The number of sequence set records: S = H / (D X C)

- This calculation is really the number of used control areas. The number of sequence set records must be the same as the number of used CAs.

  The number of index set records: I = T - S

Free space has no effect, so do not spend time trying to tune this.

Imbed and replicate can be specified, but these only help on reads later.

6. On a COLD start, try deleting the GCD data set and re-allocating and initializing it using JCL prior to the CICS job. This may be faster than allowing CICS to clean up the GCD at COLD start. CICS deletes all entries it finds in the GCD, resulting in a lot of processing.

   Note: If you do reinitialize the GCD data set (for any reason), you must also do the same for the LCD data set. The reverse also applies. See the *CICS/ESA System Definition Guide* for more information.

7. Allocate your DATA and INDEX data sets on different units, if possible.
8. Consider the use of autoinstalled terminals as a way of improving cold start, even if you do not expect any storage savings. On startup, fewer terminals are installed, thereby reducing the startup time.
9. The RAPOOL system initialization parameter should be set to a value that allows faster autoinstall rates. For a discussion of this, see “Receive-any pool (RAPOOL)” on page 196.
10. Specify the buffer, string, and key length parameters in the SHRCTL macro in the FCT. This reduces the time taken to build the LSR pool, and also reduces the open time for the first file to use the pool.
11. If you have defined performance groups for the CICS system, ensure that all steps preceding the CICS step are also in the same performance group or, at least, have a high enough dispatching priority so as not to delay their execution.
12. The use of DISP=(...,PASS) on any non-VSAM data set used in steps preceding CICS reduces allocation time the next time they are needed. If you do not use PASS on the DD statement, this causes the subsequent allocation of these data sets to go back through the catalog: a time-consuming process.
13. If possible, have one VSAM user catalog with all of the CICS VSAM data sets and use a STEPCAT DD statement to reduce the catalog search time.
14. Keep the number of libraries defined by DFHRPL to a minimum. One large library requires less time to perform the LLACOPY than many smaller libraries.
15. Use of the shared modules in the link pack area (LPA) can help to reduce the time to load the CICS nucleus modules. See the *CICS/ESA Installation Guide* for advice on how to install CICS modules in the LPA.

16. CICS does not load programs at startup time for resident programs. The storage area is reserved, but the program is actually loaded on the first access through program control for that program. This speeds startup. The correct way to find a particular program or table in storage is to use the program-control LOAD facility to find the address of the program or table. The use of the LOAD facility physically loads the program into its predefined storage location if it is the first access.

    The use of a PLTPI task to load these programs is one possible technique, but you must bear in mind that the CICS system is not operational until the PLTPI processing is complete, so you should not load every program. Load only what is necessary, or the startup time will appear to increase.

17. Use automatic journal archiving or allow the definitions of the system log and other journals in the JCT to default to the LRU (least recently used) option. This enables CICS to write to the less recently used data set while you archive the more recently used data set. This avoids a delay in CICS online operation.

CI size of data and index components of the GCD

General VSAM tuning guidelines for the NSR environment apply to tuning for the GCD. 4KB or 8KB for the data CI, and 2KB for the index CI, should be appropriate in most cases. Consult the VSAM manuals for special considerations such as different DASD device types.

Buffer considerations

To achieve a balance between virtual storage usage and the time for a CICS warm shutdown, a specification of BUFND=4 and BUFNI=3 plus ((level of indexes) – 1) is recommended.

For example, if the GCD has a two-level index, the JCL statement should be as follows:

```
//DFHGCD DD DSN=CICS330.DFHGCD,DISP=SHR,AMP=('BUFND=4,BUFNI=4')
```

The number of index levels can be obtained by using the IDCAMS LISTCAT command against a GCD after CICS has been shut down. Because cold start mainly uses sequential processing, it should not require any extra buffers over those automatically allocated when CICS opens the file.

You may wish to increase the number of buffers to improve autoinstall performance. The minimum you should specify is the number suggested above for warm shutdown. This should stop the high-level index being read for each autoinstall.

| Note that if you have a large number of devices autoinstalled, shutdown can fail due to the MXT system initialization parameter being reached or CICS becoming short on storage. To prevent this possible cause of shutdown failure, you should consider putting the CATD transaction in a class of its own to limit the number of concurrent CATD transactions. Also, AIQMAX can be specified to limit the number of devices that can be queued for autoinstall. This protects against abnormal consumption of virtual storage by the autoinstall process, caused as a result of some other abnormal event.

If this limit is reached, the AIQMAX system initialization parameter affects the LOGON, LOGOFF and BIND processing by CICS. CICS requests VTAM to stop passing such requests to CICS. VTAM holds the requests until CICS indicates that it can accept further commands (this occurs when CICS has processed a queued autoinstall request).

Part 5. Appendixes

The four appendixes are:

Appendix A. CICS statistics tables

Product-Sensitive programming interface

This appendix provides reference information about CICS statistics. For information about the interpretation of CICS statistics see Chapter 5, “Using CICS statistics” on page 41.

Interpreting CICS statistics

All five types of CICS statistics record (interval, end-of-day, requested, requested reset, and unsolicited) present information as SMF records. The numbers used to identify each SMF statistics record are given in the copybook DFHSTIDS. Programming information about the formats of CICS statistics records are given in the *CICS/ESA Customization Guide*.

Each area of CICS statistics is listed below in the following format:

Statistics area: Statistics type

Brief description, if appropriate.

Name of the assembler DSECT mapping this data.

DFHSTUP name	Field name	Description
DFHSTUP name is the name as it appears on the DFHSTUP report.	**Field name** is the name as it appears in the DSECT referred to above.	**Description** is a brief description of the statistics field. Reset characteristic: Reset characteristic of the statistics field at a statistics interval collection. The values can be: • Not reset • Reset to zero • Reset to 1 • Reset to current values (this applies to peak values only) • An exception to the above (these will be documented).

Statistics areas are listed alphabetically.

Summary report: The Statistics Utility Program (STUP) provides a summary report facility that can be selected using a DFHSTUP control parameter. Information on how to run DFHSTUP is given in the *CICS/ESA Operations and Utilities Guide*. When selected, the summary report is placed after all other reports. The DFHSTUP summary report facility summarizes (totals, peaks, and averages) the interval, unsolicited, requested reset and end-of-day statistics on an “applid” by “applid” basis. Requested statistics are not involved in the production of the summary report.

The summary report feature uses all of the appropriate statistic collections contained on the SMF data set. Therefore, depending on when the summary report feature is executed and when the SMF data set was last cleared, summary reports may be produced covering an hour, a week, or any desired period of time. Note that due to the potential magnitude of the summary data, it is not recommended that a summary period extend beyond one year.

Within each of the following sections, the meaning of the summary statistics is given. Because the summary statistics are computed offline by the DFHSTUP utility, the summary statistics are not available to online users. Due to the potential magnitude of the summary data, and due to limited page width, summary data may be represented as a scaled value. For example, if the total number of terminal input messages is 1234567890, this value is shown as 1234M, where 'M' represents millions. Other scaling factors used are 'B' for billions and 'T' for trillions. Scaling is only performed when the value exceeds 99999999, and only then when page width is limited, for example in terminal statistics.

Table 13. Statistics listed in this appendix

Autoinstall global statistics

This is the DFHSTUP listing for terminals that are connected, while the system is running, by means of the autoinstall facility. These statistics are obtained as **interval**, **end-of-day**, or **requested** statistics. CICS also records **unsolicited** autoinstall statistics, which DFHSTUP prints in a separate report; see “Autoinstalled terminals” on page 335.

Autoinstall: Local definition statistics

These statistics are available online, and are mapped by the DFHA04DS DSECT.

DFHSTUP name	Field name	Description
Autoinstall attempts	A04VADAT	is the number of eligible autoinstall attempts made during the current session of CICS to create terminal entries as users logged on. For an attempt to be considered eligible, CICS and VTAM must not be terminating, autoinstall must be enabled, and the terminal type must be valid for autoinstall (not pipeline, LU6.1, or LU6.2 parallel sessions).
		Reset characteristic: reset to zero
Rejected attempts	A04VADRJ	is the number of eligible autoinstall attempts that were subsequently rejected during the current session of CICS. Reasons for rejection can be maximum concurrency value exceeded, invalid bind, the user program has rejected the logon, and so on. If this number is unduly high, check the reasons for rejection.
		Reset characteristic: reset to zero
Deleted attempts	A04VADLO	is the number of deletions of terminal entries as users logged off during the current session.
		Reset characteristic: reset to zero
Peak concurrent attempts	A04VADPK	is the highest number of attempts made during the current session to create terminal entries as users logged on at the same time.
		Reset characteristic: reset to current value
Times the peak was reached	A04VADPX	is the number of times when the highest number of attempts were made during the current session to create terminal entries as users logged on at the same time.
		Reset characteristic: reset to 1
Times SETLOGON HOLD issued	A04VADSH	is the number of times that the SETLOGON HOLD command was issued during this run of CICS. CICS issues the VTAM SETLOGON HOLD command when the maximum number of concurrent autoinstall requests allowed (the AIQMAX= system initialization parameter) is exceeded.
		Reset characteristic: reset to zero
Queued logons	A04VADQT	is the number of attempts that were queued for logon due to delete in progress of the TCTTE for the previous session with the same LU.
		Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Peak of queued logons	A04VADQK	is the highest number of logons that were queued waiting for TCTTE deletion at any one time. If this is unduly high, consider increasing the delete delay interval parameter of the AILDELAY system initialization parameter.
		Reset characteristic: reset to current value
Times queued peak reached	A04VADQX	is the number of times this peak was reached.
		Reset characteristic: reset to 1

Autoinstall: remote definitions - shipped terminal statistics

DFHSTUP name	Field name	Description
Remote delete interval	A04RDINT	is the currently-specified time delay, in the form **hhmmss**, between invocations of the timeout delete transaction that removes redundant shipped terminal definitions. The value is set either by the DSHIPINT system initialization parameter, or by a subsequent SET DELETSHIPPED command.
		Reset characteristic: not reset
Remote delete idle time	A04RDIDL	is the currently-specified minimum time, in the form **hhmmss**, that an inactive shipped terminal definition must remain installed in this region, before it becomes eligible for removal by the CICS timeout delete transaction. The value is set either by the DSHIPIDL system initialization parameter, or by a subsequent SET DELETSHIPPED command.
		Reset characteristic: not reset
Shipped terminals built	A04SKBLT	is the number of shipped remote terminal definitions installed at the start of the recording period, plus the number built during the recording period. This value equates to the sum of “Shipped terminals installed” and “Shipped terminals deleted”.
		Reset characteristic: reset to number of skeletons installed
Shipped terminals installed	A04SKINS	is the number of shipped remote terminal definitions currently installed in this region.
		Reset characteristic: not reset
Shipped terminals deleted	A04SKDEL	is the number of shipped remote terminal definitions deleted during the recording period by the TIMEOUT transaction.
		Reset characteristic: reset to zero
Times interval expired	A04TIEXP	is the number of times the remote delete interval (A04RDINT) expired since the start of the recording period.
		Reset characteristic: reset to zero
Remote deletes received	A04RDREC	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions received by this region since the start of the recording period.
		Reset characteristic: reset to zero
Remote deletes issued	A04RDISS	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions issued by this region since the start of the recording period.
		Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Successful remote deletes	A04RDDEL	is the number of shipped terminal definitions deleted from this region because of old-style remote delete instructions, since the start of the recording period.
		Reset characteristic: reset to zero
Total idle count	A04TIDCT	is the total number of times that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse.
		This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDCT).
		Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	A04TIDLE	is the total time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse.
		This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDLE).
		Reset characteristic: reset to zero
Average idle time		is the average idle time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse.
		This number does not include the remote terminal definitions currently idle awaiting reuse.
		This value is calculated offline by DFHSTUP and is, therefore, not accessible through the EXEC CICS COLLECT STATISTICS command.
		Reset characteristic: not reset
Maximum idle time	A04TMAXI	is the maximum time (expressed in STCK units) for which a previously idle shipped terminal definition had been idle during the recording period.
		This number does not include the remote terminal definitions currently idle awaiting reuse (A04CMAXI).
		Reset characteristic: reset to current value
NOT IN THE DFHSTUP REPORT	A04CIDCT	Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	A04CIDLE	Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	A04CMAXI	Reset characteristic: Not reset

Autoinstall: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Autoinstall attempts	is the total number of eligible autoinstall attempts made during the entire CICS session to create terminal entries as users logged on. For an attempt to be considered eligible, CICS and VTAM must not be terminating, autoinstall must be enabled, and the terminal type must be valid for autoinstall (not pipeline, LU6.1, or LU6.2 parallel sessions).
Rejected attempts	is the total number of eligible autoinstall attempts that were subsequently rejected during the entire CICS session. Reasons for rejection can be maximum concurrency value exceeded, invalid bind, the user program has rejected the logon, and so on. If this number is unduly high, check the reasons for rejection.
Deleted attempts	is the total number of deletions of terminal entries as users logged off during the entire session.
Peak concurrent attempts	is the highest number of attempts made during the entire CICS session to create terminal entries as users logged on at the same time.
Times the peak was reached	is the number of times that the “peak concurrent attempts” value was reached during the entire CICS session.
Times SETLOGON HOLD issued	is the number of times that the SETLOGON HOLD command was issued during the entire run of CICS. CICS issues the VTAM SETLOGON HOLD command when the maximum number of concurrent autoinstall requests allowed (the AIQMAX= system initialization parameter) is exceeded.
Queued logons	is the total number of attempts that were queued for logon due to delete in progress of the TCTTE for the previous session with the same LU.
Peak of queued logons	is the highest number of logons that were queued waiting for TCTTE deletion at any one time. If this is unduly high, consider increasing the delete delay interval parameter of the AILDELAY system initialization parameter.
Times queued peak reached	is the number of times that the “peak of queued logons” value was reached.
Remote delete interval	is the currently-specified time delay, in the form **hhmmss**, between invocations of the timeout delete transaction that removes redundant shipped terminal definitions. The value is set either by the DSHIPINT system initialization parameter, or by a subsequent SET DELETSHIPPED command.
Remote delete idle time	is the currently-specified minimum time, in the form **hhmmss**, that an inactive shipped terminal definition must remain installed in this region, before it becomes eligible for removal by the CICS timeout delete transaction. The value is set either by the DSHIPIDL system initialization parameter, or by a subsequent SET DELETSHIPPED command.
Shipped terminals built	is the number of shipped remote terminal definitions installed at the start of the recording period, plus the number built during the recording period. This value equates to the sum of “Shipped terminals installed” and “Shipped terminals deleted”.
Shipped terminals installed	is the number of shipped remote terminal definitions currently installed in this region.
Shipped terminals deleted	is the number of shipped remote terminal definitions deleted during the recording period by the TIMEOUT transaction.
Times interval expired	is the number of times the remote delete interval (A04RDINT) expired since the start of the recording period.
Remote deletes received	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions received by this region since the start of the recording period.
Remote deletes issued	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions issued by this region since the start of the recording period.

DFHSTUP name	Description
Successful remote deletes	is the number of shipped terminal definitions deleted from this region because of old-style remote delete instructions, since the start of the recording period.
Total idle count	is the total number of times that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse. This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDCT).
Total average idle time	is the average idle time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse. This number does not include the remote terminal definitions currently idle awaiting reuse.
Maximum idle time	is the maximum time (expressed in STCK units) for which a previously idle shipped terminal definition had been idle during the recording period. This number does not include the remote terminal definitions currently idle awaiting reuse (A04CMAXI).

Autoinstalled terminals

The autoinstalled terminal statistics are of the **unsolicited** type only. These unsolicited statistics are produced when an autoinstalled terminal is about to be deleted due to the terminal being logged off. It contains the history of the activity on that terminal since the last interval. On a statistics interval, currently logged-on autoinstalled terminals are reported under **Terminal control: Resource statistics**, mapped by DSECT DFHA06DS. The statistics for an autoinstalled terminal are reset at that time. For total activity from logon to logoff, consult the summary report for this terminal.

This category of statistics is not related directly to autoinstall global statistics, which are described on page 331.

Autoinstalled terminal statistics: Unsolicited

These statistics are available online, and are mapped by the DFHAUSDS DSECT.

DFHSTUP name	Field name	Description
Terminal identifier	AUSTETI	is a unique identifier for the terminal. Reset characteristic: not reset
Input messages	AUSTENI	is the number of input messages received from the terminal. Reset characteristic: reset to zero
Output messages	AUSTENO	is the number of output messages sent to the terminal. Reset characteristic: reset to zero
Transactions	AUSTEOT	is the number of transactions that the terminal ran during the time it was logged on. Reset characteristic: reset to zero
Transmission errors	AUSTETE	is the number of transmission errors which occurred during the time that the terminal was logged on. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Transaction errors	AUSTEOE	is the number of transaction errors which occurred during the time that the terminal was logged on. Reset characteristic: reset to zero.
Terminal Luname	AUSLUNAM	is the terminal Luname. Reset characteristic: not reset
Logon Time	AUSONTM	is the time at which this terminal was autoinstalled. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK) value in local time. Reset characteristic: not reset
Logoff Time	AUSOFFTM	is the time at which this terminal was logged-off. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK) value in local time. Reset characteristic: not reset
Logon Duration	n/a	is the logged on duration for this terminal. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. This field does not appear in the Dsect; is calculated as the difference between AUSOFFTM and AUSONTM. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	AUSGONTM	is the time at which this terminal was autoinstalled. This time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK) value in GMT. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	AUSGOFTM	is the time at which this terminal was logged-off. This time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK) value in GMT. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	AUSPRTY	is the terminal priority. Reset characteristic: not reset

Autoinstalled: Summary terminal statistics: Unsolicited

Summary statistics are not available online.

DFHSTUP name	Description
Terminal identifier	is a unique identifier for the terminal.
Input messages	is the total number of input messages received from the terminal.
Output messages	is the total number of output messages sent to the terminal.
Transactions	is the total number of transactions that the terminal ran during the time it was logged on.
Transmission errors	is the total number of transmission errors which occurred during the time that the terminal was logged on.
Transaction errors	is the total number of transaction errors which occurred during the time that the terminal was logged on.

DFHSTUP name	Description
Ave Logged On Time	is the average time that the terminal was logged on.
	APAR PN91596
	MJO 23/1/97
	This field is not totalled in the DFHSTUP report.

DBCTL session termination statistics

DBCTL statistics are of the **unsolicited** type only. They appear on a separate report to the other types of CICS statistics.

The DBCTL statistics exit DFHDBSTX is invoked by the CICS adapter (DFHDBAT), and CICS statistics information is collected by the statistics domain whenever DBCTL is disconnected as a result of:

- An orderly or immediate disconnection of the DBCTL using the menu transaction CDBC
- An orderly termination of CICS.

Note: If there is an immediate shutdown or abend of CICS, the latest CICS-DBCTL session statistics are lost. The function of DFHDBSTX is to invoke the statistics domain supplying the data that has been returned from the database resource adapter (DRA) relating to the individual CICS-DBCTL session.

CICS termination statistics that contain the number of DL/I calls by type, issued against each DL/I database, are not produced by CICS in the DBCTL environment. DBCTL produces this type of information.

For more information about CICS-DBCTL statistics, see the *CICS/ESA CICS-IMS Database Control Guide*.

DBCTL: Global statistics

These statistics are mapped by the DFHDBUDS DSECT.

DFHSTUP name	Field name	Description
CICS DBCTL session number	STADSENO	is the number of the CICS-DBCTL session and is incremented every time you connect and disconnect.
		Reset characteristic: not reset
DBCTL identifier	STATDBID	is the name of the DBCTL session.
		Reset characteristic: not reset
DBCTL RSE name	STARSEN	is the name of the DBCTL recoverable service element (RSE).
		Reset characteristic: not reset
Time CICS connected to DBCTL	STALCTIM	is the time when CICS was connected to DBCTL. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at local time; however, the DSECT field contains the time as a local store clock (STCK) value.
		Reset characteristic: not reset

DFHSTUP name	Field name	Description
Time CICS disconnected from DBCTL	STALDTIM	is the time when CICS was disconnected from DBCTL. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at local time; however, the DSECT field contains the time as a local store clock (STCK) value.
		Reset characteristic: not reset
NOT IN DFHSTUP REPORT	STACTIME	is the time when CICS was connected to DBCTL. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at GMT; however, the DSECT field contains the time as a GMT store clock (STCK) value.
		Reset characteristic: not reset
NOT IN DFHSTUP REPORT	STADTIME	is the time when CICS was disconnected from DBCTL. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at GMT; however, the DSECT field contains the time as a GMT store clock (STCK) value.
		Reset characteristic: not reset
Minimum number of threads	STAMITHD	is the minimum value specified in the DRA startup parameter table.
		Reset characteristic: not reset
Maximum number of threads	STAMATHD	is the maximum value specified in the DRA startup parameter table.
		Reset characteristic: not reset
Times minimum threads hit	STANOMITHD	is the number of times the CICS-DBCTL session has hit the minimum thread value.
		Reset characteristic: not reset
Times maximum threads hit	STANOMATHD	is the number of times the CICS-DBCTL session has hit the maximum thread value.
		Reset characteristic: not reset
Elapsed time at maximum threads	STAELMAX	is the elapsed time, expressed as *hours:minutes:seconds.decimals*, for which the CICS-DBCTL session is running at the maximum thread value.
		Reset characteristic: none
Peak number of threads	STAHIWAT	is the highest number of threads used throughout the CICS-DBCTL session.
		Reset characteristic: not reset
Successful PSB schedules	STAPSBSU	is the number of times the CICS-DBCTL session has successfully scheduled a program specification block (PSB).
		Reset characteristic: not reset

DBCTL: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
CICS DBCTL session number	is the number of the CICS-DBCTL session.
DBCTL identifier	is the name of the DBCTL session.
DBCTL RSE name	is the name of the DBCTL recoverable service element (RSE).
Minimum number of threads	is the minimum value specified in the DRA startup parameter table.

DFHSTUP name	Description
Maximum number of threads	is the maximum value specified in the DRA startup parameter table.
Times minimum threads hit	is the total number of times the CICS-DBCTL session has hit the minimum thread value.
Times maximum threads hit	is the total number of times the CICS-DBCTL session has hit the maximum thread value.
Elapsed time at maximum threads	is the elapsed time, expressed as *days–hours:minutes:seconds.decimals*, for which the CICS-DBCTL session is running at the maximum thread value.
Peak number of threads	is the highest number of threads used throughout the CICS-DBCTL session.
Successful PSB schedules	is the total number of times the CICS-DBCTL session has successfully scheduled a program specification block (PSB).

Dispatcher statistics

Dispatcher domain: Global statistics

These statistics are available online, and are mapped by the DFHDSGDS DSECT.

DFHSTUP name	Field name	Description
Start time	DSGLSTRT	is the time at which the dispatcher started. This value can be used as an approximate time at which CICS started. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value in local time. Reset characteristic: not reset
NOT IN DFHSTUP REPORT	DSGSTART	is the time at which the dispatcher started. This value can be used as an approximate time at which CICS started. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value in GMT. Reset characteristic: not reset
Current number of tasks	DSGCNT	is the current number of tasks in the system. This figure includes all system tasks and all user tasks. Reset characteristic: not reset
Peak number of tasks	DSGPNT	is the peak value of the number of tasks concurrently in the system. Reset characteristic: reset to current value
Current ICV time (msec)	DSGICVT	is the ICV time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Current ICVTSD time (msec)	DSGICVSD	is the ICVTSD time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	DSGASIZE	records the current number of MVS task control blocks (TCBs) in the system under which the CICS dispatcher runs. Reset characteristic: not reset

Dispatcher domain: TCB statistics

The following fields are mapped by the DSGTCB DSECT within the DFHDSGDS DSECT. The DSGTCB DSECT is repeated for each mode of TCB in CICS (DSGASIZE). The TCB statistics only have meaning if the TCB is active (DSGTCBF1). There are four modes of TCB:

1. TCB 1 is the quasi-reentrant TCB.
2. TCB 2 is the resource owning TCB.
3. TCB 3 is the concurrent TCB. It is controlled by the system initialization parameter, SUBTSKS=0|1.
4. TCB 4 is the secondary LU TCB. It is only present if the system initialization parameter FEPI=YES is specified.

DFHSTUP name	Field name	Description
Mode	NOT IN THE DSECT	contains, in the DFHSTUP report, either QUASI, RESOURCE, CO (concurrent), or SZ (fourth TCB), depending upon which TCB it refers to. Reset characteristic: none
NOT IN THE DFHSTUP REPORT	DSGTCBF1	TCB flag byte X'80' means the TCB is active. If the TCB is not active there are no statistics in the following fields. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	DSGTCBNM	name of the TCB that the following statistics refer to. The names may be 'CO_SUBD', 'QR_SUBD', 'RO_SUBD', and 'SZ_SUBD'. Reset characteristic: not reset
MVS Waits	DSGSYSW	is the number of MVS waits which occurred on this TCB. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	DSGPERCT	is percentage use of the processor for this TCB. This value is continuously recalculated and does not represent the percentage processor time for the current statistics interval. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Accum Time in MVS wait	DSGTWT	is the accumulated real time that the CICS region was in an MVS wait, that is, the total time used between an MVS wait issued by the dispatcher and the return from the MVS wait. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Accum Time Dispatched	DSGTDT	is the accumulated real time that this TCB has been dispatched by MVS, that is, the total time used between an MVS wait issued by the dispatcher and the subsequent wait issued by the dispatcher. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	DSGTCT	is the accumulated CPU time taken for this DS task, that is, the processor time used by this TCB while executing the default dispatcher task (DSTCB). The DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Accum CPU Time / TCB	DSGACT	is the accumulated CPU time taken for this TCB, that is, the total time that this TCB has been in execution. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero

Dispatcher domain: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Start time	is the time at which the dispatcher started. This value can be used as an approximate time at which CICS started. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at the local time); however, the DSECT field contains the time as a local store clock (STCK) value.
Average number of tasks	is the average number of tasks in the system. This figure is calculated by averaging the “current number of tasks” statistic as reported in the interval report. Hence, the larger the statistics interval the more accurate this figure will be.
Peak number of tasks	is the peak number of tasks concurrently in the system.
Peak ICV time (msec)	is the peak ICV time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands.
Peak ICVTSD time (msec)	is the peak ICVTSD time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands.

Dispatcher domain: Summary TCB statistics

The following statistics are repeated for each of the four modes of TCB:

1. TCB 1 is the quasi-reentrant TCB.
2. TCB 2 is the resource owning TCB.
3. TCB 3 is optional (It is controlled by the system initialization parameter, SUBTSKS=0|1 and is the concurrent TCB.
4. TCB 4 is optional (It is controlled by the system initialization parameter, FEPI) and is the secondary LU mode.

DFHSTUP name	Description
Mode	Contains, in the DFHSTUP report, either QUASI, RESOURCE, CO (concurrent), or SZ (fourth TCB), depending upon which TCB it refers to.
MVS Waits	is the total number of MVS waits which occurred on this TCB.
Total Time in MVS wait	is the total real time that this TCB was in an MVS wait. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Total Time Dispatched	is the total real time that this TCB has been dispatched by MVS. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Total CPU Time / TCB	is the total CPU time taken for this TCB. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*. There is only a non-zero value if performance class monitoring is active when using an MCT with CPU=YES specified.

DL/I: Global statistics

The following statistics fields are available for a CICS system with DL/I. They are **not** available online, and are mapped by the DFHA25DS DSECT.

DFHSTUP name	Field name	Description
Peak DMB pool usage	A25DMBMU	is the peak number of bytes used concurrently from the DMB pool during the entire CICS session.
		Reset characteristic: reset to current value
– from a pool size of	A25DMBMS	is the maximum number of bytes in the DMB pool, specified in the DMBPL parameter of the SIT. This value is expressed in KB.
		Reset characteristic: not reset
Peak PSB pool usage	A25PSBMU	is the peak number of bytes used concurrently from the PSB pool during the entire CICS session.
		Reset characteristic: reset to current value
– from a pool size of	A25PSBMS	is the maximum number of bytes in the PSB pool, specified in the PSBPL parameter of the SIT. This value is expressed in KB.
		Reset characteristic: not reset
Peak ENQ pool usage	A25ENQMU	is the number of bytes used from the ENQ pool during the entire CICS session.
		Reset characteristic: not reset
– from a pool size of	A25ENQMS	is the maximum number of bytes in the ENQ pool, specified in the ENQPL parameter of the SIT. This value is expressed in KB.
		Reset characteristic: not reset

DFHSTUP name	Field name	Description
Peak concurrent DL/I threads	A25THRDM	is the peak number of DL/I threads used concurrently during the entire CICS session. Reset characteristic: reset to current value
– from a maximum of	A25THRDS	is the maximum number of threads specified in the DLTHRED parameter of the SIT. Reset characteristic: not reset
Active DL/I threads	A25THRDA	is the number of DL/I threads in use when the statistics request was issued. Reset characteristic: not reset
Waits for DMB pool space	A25DMBEX	is the number of times a DMB pool request exceeded the number of bytes available in the pool. Thus, it is a count of how often the IMS/ESA routine is called to free up the least-used buffers. Reset characteristic: reset to zero

Dump statistics

The dump domain collects global and resource statistics for both system and transaction dumps which occur during the CICS run.

System dumps

Dump domain: System dump global statistics

These statistics fields contain the global data collected by the dump domain for system dumps. They are available online, and are mapped by the DFHSDGDS DSECT.

DFHSTUP name	Field name	Description
Dumps taken	SYS_DUMPS_TAKEN	is the number of system dumps taken by the whole system during the present run of CICS. This number does not include suppressed dumps. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request. Reset characteristic: reset to zero
Dumps suppressed	SYS_DUMPS_SUPPR	is the number of system dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: • A user exit • The dump table • A global system dump suppression. Reset characteristic: reset to zero

Dump domain: System dump resource statistics

These statistics fields contain the data collected by the dump domain for system dumps, by dump code. They are available online, and are mapped by the DFHSDRDS DSECT.

DFHSTUP name	Field name	Description
Dumpcode	SDRCODE	is the system dump code. This code is a CICS message number with the DFH prefix and the action code suffix (if any) removed. For guidance information about CICS messages, see the *CICS/ESA Messages and Codes* manual. Reset characteristic: not reset
Dumps	SDRSTKN	is the number of system dumps taken for the dump code identified in the Dumpcode (SDRCODE) field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request. Reset characteristic: reset to zero
Dumps suppressed	SDRSSUPR	is the number of system dumps, for the dump code identified in the Dumpcode (SDRCODE) field, which were suppressed by one of: • A user exit • The dump table • A global system dump suppression. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	SDRTTKN & SDRTSUPR	These fields are always zero. They exist here only for compatibility with the transaction dump statistics record format. A transaction dump can force a system dump to be taken as well (it is an option in the transaction dump table), but a system dump cannot force a transaction dump to be taken. Reset characteristic: not applicable

Dump domain: Summary system dump global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Dumps taken	is the total number of system dumps taken by the whole system during the entire run of CICS. This number does not include suppressed dumps. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.
Dumps suppressed	is the total number of system dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: • A user exit • The dump table • A global system dump suppression.

Dump domain: Summary system dump resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Dumpcode	is the system dump code. This code is a CICS message number with the DFH prefix and the action code suffix (if any) removed. For guidance information about CICS messages, see the *CICS/ESA Messages and Codes* manual.
Dumps	is the total number of system dumps taken for the dump code identified in the Dumpcode field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.
Dumps suppressed	is the total number of system dumps, for the dump code identified in the Dumpcode field, which were suppressed by one of: • A user exit • The dump table • A global system dump suppression.

Transaction dumps

Dump domain: Transaction dump global statistics

These statistics fields contain the global data collected by the dump domain for transaction dumps. They are available online, and are mapped by the DFHTDGDS DSECT.

DFHSTUP name	Field name	Description
Dumps taken	TRANS_DUMP_TAKEN	is the number of transaction dumps taken by the whole system during the present run of CICS. This number does not include suppressed dumps. Reset characteristic: reset to zero
Dumps suppressed	TRANS_DUMP_SUPP	is the number of transaction dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: • A user exit • The dump table. Reset characteristic: reset to zero

Dump domain: Transaction dump resource statistics

These statistics fields contain the data collected by the dump domain for transaction dumps, by dump code. They are available online, and are mapped by the DFHTDRDS DSECT.

DFHSTUP name	Field name	Description
Dumpcode	TDRCODE	is the transaction dump code. For guidance information about transaction abend codes, see the *CICS/ESA Messages and Codes* manual. Reset characteristic: not reset
Dumps	TDRTTKN	is the number of transaction dumps taken for the dump code identified in the Dumpcode (TDRCODE) field. Reset characteristic: reset to zero
Dumps suppressed	TDRTSUPR	is the number of transaction dumps suppressed for the dump code identified in the Dumpcode (TDRCODE) field. Reset characteristic: reset to zero
System dumps	TDRSTKN	is the number of system dumps forced by the transaction dump identified in the Dumpcode (TDRCODE) field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
System dumps suppressed	TDRSSUPR	is the number of system dumps, forced by the transaction dump identified in the Dumpcode (TDRCODE) field, which were suppressed by one of: • A user exit • The transaction dump table • A global system dump suppression. Reset characteristic: reset to zero

Dump domain: Summary transaction dump global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Dumps taken	is the total number of transaction dumps taken by the whole system during the entire run of CICS. This number does not include suppressed dumps.
Dumps suppressed	is the total number of transaction dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: • A user exit • The dump table.

Dump domain: Summary transaction dump resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Dumpcode	is the transaction dump code. For guidance information about transaction abend codes, see the *CICS/ESA Messages and Codes* manual.
Dumps	is the total number of transaction dumps taken for the dump code identified in the Dumpcode field.
Dumps suppressed	is the total number of transaction dumps suppressed for the dump code identified in the Dumpcode field.
System dumps	is the total number of system dumps forced by the transaction dump identified in the Dumpcode field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.
System dumps suppressed	is the total number of system dumps, forced by the transaction dump identified in the Dumpcode field, which were suppressed by one of: • A user exit • The transaction dump table • A global system dump suppression.

Dynamic transaction backout statistics

These statistics are available online, and are mapped by the DFHA05DS DSECT.

DFHSTUP name	Field name	Description
Number of records logged	A05DBLA	is the number of records written to the dynamic log, so that they are available for transaction backout.
		Reset characteristic: reset to zero
Number of records spilled	A05DBSA	is the number of times the dynamic log overflow mechanism had to be used.
		Reset characteristic: reset to zero

Dynamic transaction backout: Summary statistics

Summary statistics are not available online.

DFHSTUP name	Description
Number of records logged	is the number of records written to the dynamic log, so that they are available for transaction backout.
Number of records spilled	is the number of times the dynamic log overflow mechanism had to be used.

Front end programming interface (FEPI) statistics

FEPI statistics contain data about the use of each FEPI pool, a target in any pool, and FEPI connection.

FEPI: Pool statistics

These statistics give information about each FEPI pool. The statistics are available online, and are mapped by the DFHA22DS DSECT.

DFHSTUP name	Field name	Description
Pool Name	A22POOL	is the FEPI pool name.
		Reset characteristic: not reset
Targets	A22TRGCT	is the current number of targets in the pool.
		Reset characteristic: not reset
Nodes	A22NDCT	is the current number of nodes in the pool.
		Reset characteristic: not reset
Available Connections		
–Current	A22CONCT	is the number of connections in the pool.
		Reset characteristic: not reset
–Peak	A22CONPK	is the peak number of connections in the pool. This field is needed because targets and nodes may be deleted between intervals.
		Reset characteristic: reset to current value (A22CONCT)
Allocates		
–Total	A22ALLOC	is the number of conversations that have been allocated from this pool.
		Reset characteristic: reset to zero
–Peak	A22PKALL	is the peak number of concurrent conversations allocated from this pool.
		Reset characteristic: reset to current value

DFHSTUP name	Field name	Description
Allocate Waits		
NOT IN THE DFHSTUP REPORT	A22WAIT	is the current number of conversations waiting to be allocated.
		Reset characteristic: not reset
–Total	A22TOTWT	is the number of conversations that had to wait to be allocated.
		Reset characteristic: reset to zero
–Peak	A22PKWT	is the peak number of conversations that had to wait to be allocated.
		Reset characteristic: reset to current value (A22WAIT)
Allocate Timeouts	A22TIOUT	is the number of conversation allocates that timed out.
		Reset characteristic: reset to zero

FEPI: Connection statistics

These statistics give information about each FEPI connection. The statistics are available online, and are mapped by the DFHA23DS DSECT.

DFHSTUP name	Field name	Description
Pool Name	A23POOL	is the FEPI pool name.
		Reset characteristic: not reset
Target Name	A23TARG	is the FEPI target name.
		Reset characteristic: not reset
Node Name	A23NODE	is the FEPI node.
		Reset characteristic: not reset
Acquires	A23ACQ	is the number of times the connection was acquired.
		Reset characteristic: reset to zero
Conversations	A23CNV	is the number of conversations that have used this connection.
		Reset characteristic: reset to zero
Unsolicited Inputs	A23USI	is the number of times unsolicited input was received on this connection.
		Reset characteristic: reset to zero
Characters		
–Sent	A23CHOUT	is the number of characters of data sent on this connection.
		Reset characteristic: reset to zero
–Received	A23CHIN	is the number of characters of data received on this connection.
		Reset characteristic: reset to zero
Receive Timeouts	A23RTOUT	is the number of times a FEPI RECEIVE timed-out on this connection.
		Reset characteristic: reset to zero
Error Conditions	A23ERROR	is the number of VTAM error conditions raised for this connection.
		Reset characteristic: reset to zero

FEPI: Target Statistics

These statistics give information about a particular target in a pool. The statistics are available online, and are mapped by the DFHA24DS DSECT.

DFHSTUP name	Field name	Description
Target Name	A24TARG	is the FEPI target name.
		Reset characteristic: not reset
Pool Name	A24POOL	is the FEPI pool name.
		Reset characteristic: not reset
Applid	A24APPL	is the VTAM applid of the target.
		Reset characteristic: not reset
Nodes	A24NDCT	is the number of nodes connected to this target.
		Reset characteristic: not reset
Allocates	A24ALLOC	is the number of conversations **specifically** allocated to this target in this pool.
		Reset characteristic: reset to zero
Allocate Waits		
–Total	A24TOTWT	is the number of conversations that had to wait to be allocated to this target in this pool.
		Reset characteristic: reset to zero
–Wait	A24WAIT	is the number of current conversations waiting to be allocated to this target in this pool
		Reset characteristic: reset to zero
–Peak	A24PKWT	is the peak number of conversations that had to wait to be allocated to this target in this pool.
		Reset characteristic: reset to current value (A24WAIT)
Allocate Timeouts	A24TIOUT	is the number of conversation allocates to this target in this pool that timed out.
		Reset characteristic: reset to zero

FEPI: Unsolicited pool statistics

Unsolicited pool statistics are produced when a pool is discarded. The statistics are mapped by the DFHA22DS DSECT. They contain the same information as the interval statistics.

FEPI: Unsolicited connection statistics

Unsolicited connection statistics are produced when a connection is destroyed. This occurs when a DELETE POOL, DISCARD NODELIST, DISCARD POOL or DISCARD TARGETLIST command is used. The statistics are mapped by the DFHA23DS DSECT. They contain the same information as the interval statistics.

FEPI: Unsolicited target statistics

Unsolicited target statistics are produced when a target is destroyed or removed from a pool. This occurs when a DELETE POOL, DISCARD POOL or DISCARD TARGETLIST command is used. The statistics are mapped by the DFHA24DS DSECT. They contain the same information as the interval statistics.

FEPI: Pool summary statistics

Summary statistics are not available online.

DFHSTUP name	Description
Pool Name	is the FEPI pool name.
Targets	is the number of targets in the pool.
Nodes	is the number of nodes in the pool.
Available Connections	
–Current	is the number of connections in the pool.
–Peak	is the highest peak number of connections in the pool.
Allocates	
–Totals	is the total number of conversations allocated from this pool.
–Peak	is the highest peak number of concurrent conversations allocated from this pool.
Allocate Waits	
–Total	is the total number of conversations that had to wait to be allocated.
–Peak	is the highest peak number of conversations that had to wait to be allocated.
Allocate Timeouts	is the total number of conversation allocates that timed out.

FEPI: Connection summary statistics

Summary statistics are not available online.

DFHSTUP name	Description
Pool Name	is the FEPI pool name.
Target Name	is the FEPI target name.
Node Name	is the FEPI node.
Acquires	is the total number of times the connection was acquired.
Conversations	is the total number of conversations that have used this connection.
Unsolicited Inputs	is the total number of times unsolicited input was received on this connection.
Characters Sent	
–Sent	is the total number of characters of data sent on this connection.
–Received	is the total number of characters of data received on this connection.
Receive Timeouts	is the total number of times a FEPI RECEIVE timed-out on this connection.
Error Conditions	is the total number of VTAM error conditions raised for this connection.

FEPI: Target summary statistics

Summary statistics are not available online.

DFHSTUP name	Description
Target Name	is the FEPI target name.
Pool Name	is the FEPI pool name.
Applid	is the VTAM applid of the target.
Nodes	is the number of nodes in the pool.
Allocates	is the total number of conversations **specifically** allocated to this target in this pool.
Allocate Waits	

DFHSTUP name	Description
–Total	is the total number of conversations that had to wait to be allocated to this target in this pool.
–Peak	is the highest peak number of conversations that had to wait to be allocated to this target in this pool.
Allocate Timeouts	is the total number of conversations allocated to this target in this pool that timed out.

File statistics

There are four sections in the DFHSTUP report for file statistics:

- Files: Resource Information ("Files: Resource information statistics" on page 352).
- Files: Requests Information ("Files: Requests information statistics" on page 355).
- Files: Data Table Requests Information ("Files: Data table Requests Information statistics" on page 356).
- Files: Performance Information ("Files: Performance information statistics" on page 359).

Unsolicited file statistics are printed in a statistics report separate from other types of CICS statistics.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHA17DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

Files: Resource information statistics

DFHSTUP name	Field name	Description
File Name	A17FNAM	is the name you specified in: • The DEFINE FILE command of resource definition online Reset characteristic: not reset
Dataset Name	A17DSNAM	is the 44-character name defining the physical data set to the system. You may have specified this in: • The DSNAME operand specified in the DEFINE FILE command of resource definition online • The operand specified in the DD DSN= operand of the CICS JCL • By dynamic allocation of a data set to a file through the use of CEMT SET FILE DSNAME or EXEC CICS SET FILE DSNAME commands. If no data set is currently allocated to the file, this field is blank. If the file is remote, no data set name is printed but the word "**remote**" is substituted for the data set name. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Base Dataset Name (If Applicable)	A17BDSNM	In the instance that the file is a VSAM PATH, this field gives the base data set name. Reset characteristic: not reset.
Dataset Type	A17DSTYP	is the data set type. • B = BDAM • E = VSAM ESDS • K = VSAM KSDS • R = VSAM RRDS • P = VSAM PATH Reset characteristic: not reset.
DT Indicator	A17DT	is a one-byte field that contains the value “R”, or “S” or “T”, or “X”, if data table statistics fields are present in the record. “R” indicates that this is a remote file for which table read and source read statistics are present. “S” indicates that the resource was not opened as a table but was able to access data from a table associated with the same data set. “T” indicates that the resource is a data table. “X” indicates that the resource has been opened with a source data set which has an associated CICS maintained data table. Reset characteristic: not reset
Time Opened	A17LOPNT	is the time at which this file was opened. If this field is not set, A17LOPNT contains the hexadecimal value X'00000000 00000000', shown in the report as “CLOSED”. If the field is set, it contains a time expressed as a store clock (STCK) value in local time. This field contains a valid time if: • The file was open at the time the statistics were taken. • This is an unsolicited statistics request due to the file being closed. Reset characteristic: not reset
Time Closed	A17LCLST	is the time at which this file was closed. If this field is not set, A17LCLST contains the hexadecimal value X'00000000 00000000', shown in the report as “OPEN”. If the field is set, it contains a time expressed as a store clock (STCK) value in local time. Reset characteristic: not reset
Remote Name	A17RNAME	The name by which this file is known in the system or region in which it is resident. Reset characteristic: not reset.
Remote Sysid	A17RSYS	When operating in an ISC or MRO environment, and the file is held by a remote system, this field specifies the system upon which the file is resident. Reset characteristic: not reset.

DFHSTUP name	Field name	Description
Lsrpool ID	A17POOL	The identity of the local shared resource pool. This value is that specified by: • The LSRPOOLID operand of the resource definition online DEFINE FILE command. 'N' means that it is not defined in an LSR pool. Reset characteristic: not reset.
NOT IN THE DFHSTUP REPORT	A17FLOC	states whether the file is defined as being local to this CICS system, or resides on a remote CICS system. The field is one byte long, and is set to "R" if remote. Reset characteristic: not reset

Note: When the source data set of a user-maintained table is closed, the "time opened" is reset to the time at which the source was closed.

Files: Summary resource information statistics

Summary statistics are not available online.

DFHSTUP name	Description
File Name	is the name you specified in: • The DEFINE FILE command of resource definition online
Dataset Name	is the 44-character name defining the physical data set to the system.
Base Dataset Name (If applicable)	In the instance that the file is a VSAM PATH, this field gives the base data set name.
Dataset Type	is the data set type. • B = BDAM • E = VSAM ESDS • K = VSAM KSDS • R = VSAM RRDS • P = VSAM PATH
DT Indicator	is a one-byte field that contains the value "R", or "S" or "T", or "X", if data table statistics fields are present in the record. "R" indicates that this is a remote file for which table read and source read statistics are present. "S" indicates that the resource was not opened as a table but was able to access data from a table associated with the same data set. "T" indicates that the resource is a data table. "X" indicates that the resource has been opened with a source data set which has an associated CICS maintained data table.
Remote Name	The name by which this file is known in the system or region in which it is resident.
Remote Sysid	When operating in an ISC or MRO environment, and the file is held by a remote system, this field specifies the system upon which the file is resident.
Lsrpool ID	The identity of the local shared resource pool. This value is that specified via: • The LSRPOOLID operand of the resource definition online DEFINE FILE command. • The TYPE=FILE, LSRPOOL operand of the DFHFCT macro. 'N' means that it is not defined in an LSR pool.

Files: Requests information statistics

The following eight items are service request statistics. They do not tell you directly how many I/O accesses are being carried out for each transaction (a single-transaction measurement is required for this). Nevertheless, by regularly totaling the service requests against individual data sets, they can enable you to anticipate data set problems when I/O activity increases.

They list the number of service requests processed against the data set. These are dependent on the type of requests that are allowed on the data set.

DFHSTUP name	Field name	Description
File Name	A17FNAM	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro. Reset characteristic: not reset
Get Requests	A17DSRD	is the number of GET requests attempted against this file. Reset characteristic: reset to zero
Get Upd Requests	A17DSGU	is the number of GET UPDATE requests attempted against this file. Reset characteristic: reset to zero
Browse Requests	A17DSBR	is the number of GETNEXT and GETPREV requests attempted against this file. Reset characteristic: reset to zero
Update Requests	A17DSWRU	is the number of PUT UPDATE requests attempted against this file. Reset characteristic: reset to zero
Add Requests	A17DSWRA	is the number of PUT requests attempted against this file. Reset characteristic: reset to zero
Delete requests		A17DSDEL or A17RMDEL is printed here
	A17DSDEL	is the number of DELETE requests attempted against this local file. Reset characteristic: reset to zero
	A17RMDEL	is the number of DELETE requests for a VSAM file in a remote system. In systems connected by a CICS intercommunication (MRO or ISC) link, the statistics recorded for the remote files are a subset of those recorded for the files on the local system. Reset characteristic: reset to zero
VSAM EXCP Requests		
–Data	A17DSXCP	A value is printed if the FCT entry has been opened and used as a VSAM KSDS during the CICS run, even if the FCT entry is not being used as a KSDS at the time of taking statistics. See notes 1 and 2.

DFHSTUP name	Field name	Description
–Index	A17DSIXP	See notes 1 and 2.
		Reset characteristic: reset to zero

Notes: The "VSAM EXCP requests" fields indicate the number of I/O operations on the file for data and index records respectively. Also, note the following points:

1. The values printed for both items relate to the FCT entry. If dynamic allocation has been used to change the physical data sets associated with an FCT entry, the value shown is an accumulation for all the data sets.
2. Take care when using these values for files participating in data set name sharing, because VSAM maintains only one count of EXCPs for all access method control blocks (ACBs) thus connected. In this situation, the value reported against each file represents the total accesses for all sharing ACBs during the period for which the file was open. (Therefore, if all files in the data set name sharing group were open for the same period, each file would have the same EXCP values reported against it, which would be the total for all the files in the group.)

Files: Summary requests information statistics

Summary statistics are not available online.

DFHSTUP name	Description
File Name	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro.
Get Requests	is the total number of GET requests issued against this file.
Get Upd Requests	is the total number of GET UPDATE requests issued against this file.
Browse Requests	is the total number of GETNEXT and GETPREV requests issued against this file.
Update Requests	is the total number of PUT UPDATE requests issued against this file.
Add Requests	is the total number of PUT requests issued against this file.
Delete Requests	is the total number of DELETE requests issued against this file.
VSAM EXCP Request	
–Data	A value is printed if the FCT entry has been opened and used as a VSAM KSDS during the CICS run. See notes 1 and 2.
–Index	See notes 1 and 2.

Notes: The previous two fields indicate the number of I/O operations on the file for data and index records respectively. Also, note the following points:

1. The values printed for both items relate to the FCT entry. If dynamic allocation has been used to change the physical data sets associated with an FCT entry, the value shown is an accumulation for all the data sets.
2. Take care when using these values for files participating in data set name sharing, because VSAM maintains only one count of EXCPs for all ACBs thus connected. In this situation, the value reported against each file represents the total accesses for all sharing ACBs during the period for which the file was open. (Therefore, if all files in the data set name sharing group were open for the same period, each file would have the same EXCP values reported against it, which would be the total for all the files in the group.)

Files: Data table Requests Information statistics

If the file is a data table, further fields are present in the statistics record. The presence of these additional fields is indicated by the value "R", or "S", or "T", or "X" in the field A17DT. Their names and meanings are as follows:

DFHSTUP name	Field name	Description
File Name	A17FNAM	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro. Reset characteristic: not reset
Close Type	A17DTTYP	This one-byte field is set to: "C" when a CICS maintained table is closed. "P" when a file which has been accessing a CICS-maintained table is closed but the table remains open because there are other files still open which are using the table, "S" when the source data set for a user table is being closed, "U" when a user maintained table is closed. Reset characteristic: not reset
Read Requests	A17DTRDS	is the number of attempts to retrieve records from the table. Reset characteristic: reset to zero
Recs¬in Table	A17DTRNF	is the number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress. Reset characteristic: reset to zero
Adds from Reads	A17DTAVR	is the number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress. Reset characteristic: reset to zero
Add Requests	A17DTADS	is the number of attempts to add records to the table as a result of WRITE requests. Reset characteristic: reset to zero
Adds rejected – Exit	A17DTARJ	is the number of records CICS attempted to add to the table which were rejected by the global user exit. Reset characteristic: reset to zero
Adds rejected – Table Full	A17DTATF	is the number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified. Reset characteristic: reset to zero
Rewrite Requests	A17DTRWS	is the number of attempts to update records in the table as a result of REWRITE requests. Reset characteristic: reset to zero
Delete Requests	A17DTDLS	is the number of attempts to delete records from the table as a result of DELETE requests. Reset characteristic: reset to zero
Highest Table Size	A17DTSHI	is the peak number of records present in the table. Reset characteristic: reset at close

DFHSTUP name	Field name	Description
Storage Alloc(K)	A17DTALT	is the total amount of storage allocated to the data table. The DFHSTUP report expresses the storage in kilobytes. DFHSTUP does not total the storage allocated for all data tables because multiple files may share the same data table. Reset characteristic: not reset

Note: The Request Information statistics output for a data table represents the activity of the source data set, and the data table request information represents the activity of the data table. Thus, for a CICS-maintained table, you would expect to find similar counts in both sections of the statistics output for requests which modify the table, because both the source data set and the table must be updated. For a user-maintained table, there should be no updating activity shown in the data table resource information.

DFHSTUP name	Field name	Description
When using the shared data tables feature the statistics records will contain the additional information as follows:		
NOT IN THE DFHSTUP REPORT	A17DTSIZ	is the current number of records in the data table. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUST	is the total amount of storage (kilobytes) in use for the data table. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTALE	is the total amount of storage (kilobytes) allocated for the record entry blocks. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUSE	is the total amount of storage (kilobytes) in use for the record entry blocks. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTALI	is the total amount of storage (kilobytes) allocated for the index. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUSI	is the total amount of storage (kilobytes) in use for the index. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTALD	is the total amount of storage (kilobytes) allocated for the record data. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUSD	is the total amount of storage (kilobytes) in use for the record data. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTRRS	is the total number of read retries, that is the number of times reads in an AOR had to be retried because the FOR changed the table during the read. A17DTRRS is **not** a count of accesses which failed because a file owning region (FOR) was updating the specific record that the AOR wished to read. In such cases, the request is function shipped and is counted in the “source reads.” Reset characteristic: not reset

Note: Data table fields are present in the statistics records but contain zeros if shared data tables are not installed or the resource is not a data table.

Files: Summary data table requests statistics

Summary statistics are not available online.

DFHSTUP name	Description
File Name	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro.
Table Type	This one-byte field is set to: "C" when a CICS maintained table is closed. "P" when a file which has been accessing a CICS maintained table is closed but the table remains open because there are other files still open which are using the table, "S" when the source data set for a user table is being closed, "U" when a user maintained table is closed.
Successful Reads	is the total number of reads from the data table.
Recs - in Table	is the total number of records in the data table.
Adds from Reads	is the total number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress.
Add Requests	is the total number of attempts to add records to the table as a result of WRITE requests.
Adds Rejected	
–Exit	is the total number of records CICS attempted to add to the table which were rejected by the global user exit.
–Table Full	is the total number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified.
Rewrite Requests	is the total number of attempts to update records in the table as a result of REWRITE requests.
Delete Requests	is the total number of attempts to delete records from the table as a result of DELETE requests.
Highest Table Size	is the peak number of records present in the table.

Files: Performance information statistics

These statistics are available online, and are mapped by the DFHA17DS DSECT.

DFHSTUP name	Field name	Description
File Name	A17FNAM	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro. Reset characteristic: not reset
Strings	A17STRNO	The maximum permissible number of concurrent updates. Reset characteristic: not reset.
Active Strings	A17DSASC	The current number of updates against the file. Reset characteristic: not reset.
Wait On Strings		
–Current	A17DSCWC	The current number of 'waits' for strings against the file. Reset characteristic: not reset

DFHSTUP name	Field name	Description
–Total	A17DSTSW	The total number of 'waits' for strings against the file. Reset characteristic: reset to zero
–Highest	A17DSHSW	The highest number of 'waits' for strings against the file. Reset characteristic: reset to current value
Buffers		
–Data	A17DSDNB	The number of buffers to be used for data. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report. Reset characteristic: not reset.
–Index	A17DSINB	The number of buffers to be used for index. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report. Reset characteristic: not reset.

Files: Summary performance information statistics

Summary statistics are not available online.

DFHSTUP name	Description
File Name	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro.
Strings	The maximum permissible number of concurrent updates.
Wait On Strings	
–Total	The total number of ‘waits’ for strings against the file.
–HWM	The highest number of 'waits' for strings against the file.
Buffers	
–Data	The number of buffers to be used for data. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report.
–Index	The number of buffers to be used for index. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report.

IRC batch statistics

The IRC batch area of the DFHSTUP listing is for a CICS system allowing the shared use of a DL/I database in the CICS region through interregion communication (IRC).

IRC batch: Global statistics

These statistics are available online, and are mapped by the DFHA19DS DSECT.

DFHSTUP name	Field name	Description
Peak concurrent batch jobs sharing database	A19EMCTH	is the peak number of batch jobs concurrently sharing the database. Reset characteristic: reset to current
Total batch jobs sharing database	A19ETOTH	is the number of batch jobs that have shared or are sharing the database. Reset characteristic: reset to zero

IRC batch: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Peak concurrent batch jobs sharing database	is the peak of the number of batch jobs concurrently sharing the database.
Total batch jobs sharing database	is the total number of batch jobs that have shared the database.

ISC/IRC system and mode entries

The ISC/IRC system and mode entry statistics area of the DFHSTUP listing is for a CICS system using intersystem communication. This provides summary statistics for the CICS intercommunication facility.

System entry

ISC/IRC system entry: Resource statistics

The system entry statistics record both ISC and IRC statistics. Some entries are unique to one or the other, and show zero activity if that function is not used. Statistics are provided for each system entry in the terminal definition.

These statistics are available online, and are mapped by the DFHA14DS DSECT. This DSECT is to be used:

- For processing data returned for an online enquiry for a connection (EXEC CICS COLLECT STATISTICS)
- For processing connection statistics offline (SMF)
- For processing the connection totals (the summation of all defined connections in this CICS region).

CICS always allocates a SEND session when sending an IRC request to another region. Either a SEND or RECEIVE session can be allocated when sending requests using LU6.1 ISC, and either a contention loser or a contention winner session can be allocated when sending requests using APPC.

In LU6.1, SEND sessions are identified as secondaries, and RECEIVE sessions are identified as primaries.

DFHSTUP name	Field name	Description
Connection name	A14CNTN	corresponds to each system entry defined by a CONNECTION definition in the CSD.
		Reset characteristic: not reset
AIDs in chain	A14EALL	is the current number of automatic initiate descriptors (AIDs) in the AID chain.
		Reset characteristic: not reset
Generic AIDs in chain	A14ESALL	is the current number of automatic initiate descriptors (AIDs) that are waiting for a session to become available to satisfy an allocate request.
		Reset characteristic: not reset
ATIs satisfied by contention losers	A14ES1	is the number of ATI requests (queued allocates) that have been satisfied by contention loser sessions (primaries for LU6.1). This is always zero for IRC system entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics.
		Reset characteristic: reset to zero
ATIs satisfied by contention winners	A14ES2	is the number of ATI requests (queued allocates) that have been satisfied by contention winner sessions (secondaries for LU6.1). This field is the total ATIs when the system entry is for IRC. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics.
		Reset characteristic: reset to zero
Peak contention losers	A14E1HWM	is the peak number of contention loser sessions (primaries for LU6.1) that were in use at any one time. For APPC, this field is zero.
		Reset characteristic: reset to current value
Peak contention winners	A14E2HWM	is the peak number of contention winner sessions (secondaries for LU6.1) that were in use at any one time. For APPC, this field is zero.
		Reset characteristic: reset to current value

Note for the following five fields: For APPC only, if an allocate request does not specify a mode group, CICS takes the first mode group within the sessions available, and the statistics for these allocates are reported against the system entry. If an allocate specifically requests a mode entry, the statistics for these allocates go into that mode entry.

Peak outstanding allocates	A14ESTAM	is the peak number of allocate requests that were queued for this system. For APPC this field is incremented only for generic allocate requests.
		Reset characteristic: reset to current value

DFHSTUP name	Field name	Description
Total number of allocates	A14ESTAS	is the number of allocate requests against this system. For APPC: • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
Queued allocates	A14ESTAQ	is the current number of queued allocate requests against this system. An allocate is queued due to a session not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use. For APPC: • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: not reset
Failed link allocates	A14ESTAF	is the number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. For APPC: • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
Failed allocates due to sessions in use	A14ESTAO	is the number of allocate requests that failed due to a session not being currently available for use. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code. For APPC only: • This field is only incremented for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
Maximum queue time (seconds)	A14EMXQT	is the MAXQTIME specified on the CONNECTION definition. This value represents the maximum time you require to process an allocate queue on this connection. If the allocate queue would take greater than this time to process then the entire queue would be purged. This value only takes effect if the QUEUELIMIT value (A14EALIM) has been reached. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Allocate queue limit	A14EALIM	is the QUEUELIMIT parameter specified on the CONNECTION definition. If this value is reached then allocates are rejected. Reset characteristic: not reset
Number of QUEUELIMIT allocates rejected	A14EALRJ	the total number of allocates rejected due to the QUEUELIMIT value (A14EALIM) being reached. Reset characteristic: reset to zero
Number of MAXQTIME allocate queue purges	A14EQPCT	is the total number of times an allocate queue has been purged due to the MAXQTIME value (A14EMXQT). A queue is purged when the total time it would take to process a queue exceeds the MAXQTIME value. Reset characteristic: reset to zero
Number of MAXQTIME allocates purged	A14EMQPC	is the total number of allocates purged due to the queue processing time exceeding the MAXQTIME value (A14EMXQT). If sessions have not been freed after this mechanism has been invoked then any subsequent allocate requests are purged and included in this statistic as the MAXQTIME purging mechanism is still in operation. Reset characteristic: reset to zero
Number of XZIQUE allocates rejected	A14EZQRJ	is the total number of allocates rejected by the XZIQUE exit. Reset characteristic: reset to zero
Number of XZIQUE allocate queue purges	A14EZQPU	is the total number of allocate queue purges that have occurred at XZIQUE request for this connection. If accessed online using the EXEC CICS COLLECT STATISTICS command, this field additionally contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
Number of XZIQUE allocates purged	A14EZQPC	is the total number of allocates purged due to XZIQUE requesting that queues should be purged (A14EZQPU) for this connection. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation. If accessed online using the EXEC CICS COLLECT STATISTICS command, this field additionally contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
Total bids sent	A14ESBID	is the total number of bids that were sent. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Current bids in progress	A14EBID	is the number of bids currently in progress. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC system entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics. Reset characteristic: not reset
Peak bids in progress	A14EBHWM	is the peak number of bids that were in progress at any one time. A bid is sent on an LU6.1 RECEIVE session only. Reset characteristic: reset to current value
File control function shipping requests	A14ESTFC	is the number of file control requests for function shipping. Reset characteristic: reset to zero
Interval control function shipping requests	A14ESTIC	is the number of interval control requests for function shipping. Reset characteristic: reset to zero
TD function shipping requests	A14ESTTD	is the number of transient data requests for function shipping. Reset characteristic: reset to zero
TS function shipping requests	A14ESTTS	is the number of temporary storage requests for function shipping. Reset characteristic: reset to zero
DL/I function shipping requests	A14ESTDL	is the number of DL/I requests for function shipping. Reset characteristic: reset to zero
Terminal sharing requests	A14ESTTC	is the number of transaction routing commands. This number is incremented on both regions when the transaction is routed, and when the terminal I/O request is routed between regions. This field is not supported for LU6.1. Reset characteristic: reset to zero

ISC/IRC system entry: Summary resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Average number of AIDs in chain	is the average number of automatic initiate descriptors (AIDs) in the AID chain.
Average number of generic AIDs in chain	is the average number of AIDs waiting for a session to become available to satisfy an allocate request.
ATIs satisfied by contention losers	is the total number of ATI requests (queued allocates) that have been satisfied by contention loser sessions (primaries for LU6.1). This is always zero for IRC system entries.
ATIs satisfied by contention winners	is the total number of ATI requests (queued allocates) that have been satisfied by contention winner sessions (secondaries for LU6.1). This field is the total ATIs when the system entry is for IRC.
Peak contention losers	is the peak number of contention loser sessions (primaries for LU6.1). that were in use at any one time. For APPC, this field is zero.
Peak contention winners	is the peak number of contention winner sessions (secondaries for LU6.1) that were in use at any one time. For APPC, this field is zero.

DFHSTUP name	Description
Note for the following five fields: For APPC only, if an allocate request does not specify a mode group, CICS takes the first mode group within the sessions available, and the statistics for these allocates are reported against the system entry. If an allocate specifically requests a mode entry, the statistics for these allocates go into that mode entry.	
Peak outstanding allocates	is the peak number of allocation requests that were queued for this system. For APPC this field contains only generic allocate requests.
Total number of allocates	is the total number of allocate requests against this system. For APPC this field contains only generic allocate requests.
Average number of queued allocates	is the average number of queued allocate requests against this system. For APPC this field is incremented only for generic allocate requests.
Failed link allocates	is the total number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. For APPC this field is incremented only for generic allocate requests.
Failed allocates due to sessions in use	is the total number of allocate requests that failed due to a session not being currently available for use. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code. For APPC this field is incremented only for generic allocate requests.
Maximum queue time (seconds)	is the last non-zero value encountered for the MAXQTIME specified on the CONNECTION definition. This value represents the maximum time you require to process an allocate queue on this connection. If the allocate queue would take greater than this time to process the entire queue would be purged. This value only takes effect if the QUEUELIMIT value has been reached.
Allocate queue limit	is the last non-zero value encountered for the QUEUELIMIT parameter specified on the CONNECTION definition. If this value is reached then allocates are rejected.
Number of QUEUELIMIT allocates rejected	is the is the total number of allocates rejected due to the QUEUELIMIT value being reached.
Number of MAXQTIME allocate queue purges	is the total number of times an allocate queue has been purged due to the MAXQTIME value. A queue is purged when the total time it would take to process a queue exceeds the MAXQTIME value.
Number of MAXQTIME allocates purged	is the total number of allocates purged due to the queue processing time exceeding the MAXQTIME value. If sessions have not been freed after this mechanism has been invoked then any subsequent allocate requests are purged and included in this statistic as the MAXQTIME purging mechanism is still in operation.
Number of XZIQUE allocates rejected	is the total number of allocates rejected by the XZIQUE exit
Number of XZIQUE allocate queue purges	is the total number of allocate queue purges that have occurred at XZIQUE request for this connection.
Number of XZIQUE allocates purged	is the total number of allocates purged due to XZIQUE requesting that queues should be purged for this connection. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation.
Total bids sent	is the total number of bids that were sent. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.
Average bids in progress	is the average number of bids in progress. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.
Peak bids in progress	is the peak number of bids that were in progress at any one time. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.

DFHSTUP name	Description
File control function shipping requests	is the total number of file control requests for function shipping.
Interval control function shipping requests	is the total number of interval control requests for function shipping.
TD function shipping requests	is the total number of transient data requests for function shipping.
TS function shipping requests	is the total number of temporary storage requests for function shipping.
DL/I function shipping requests	is the total number of DL/I requests for function shipping.
Terminal sharing requests	is the total number of transaction routing commands. This number is incremented on both regions when the transaction is routed, and when the terminal I/O request is routed between regions. This field is not supported for LU6.1.

Mode entry

The ISC/IRC system and mode entry statistics area of the DFHSTUP listing is for a CICS system using intersystem communication. This provides summary statistics for the CICS intercommunication facility.

ISC/IRC mode entry: Resource statistics

These statistics are collected only if you have an APPC connection defined in your CICS region, and they are then produced for each mode group defined in that connection. These statistics cannot be accessed online using the EXEC CICS COLLECT STATISTICS command. They are only produced for offline processing (written to SMF).

These statistics are mapped by the DFHA20DS DSECT. This DSECT is also used to map the mode entry totals records.

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A20SYSN	is the name of the APPC connection/system that owns this mode entry. It corresponds to the system entry, defined by a CONNECTION definition in the CSD.
		Reset characteristic: not reset
Mode name	A20MODE	is the mode group name related to the the intersystem connection name above (A20SYSN). This corresponds to modename in the sessions definition.
		Reset characteristic: not reset
ATIs satisfied by contention losers	A20ES1	is the number of ATI requests (queued allocates) that have been satisfied by “contention loser” sessions belonging to this mode group.
		Reset characteristic: reset to zero
ATIs satisfied by contention winners	A20ES2	is the number of ATI requests (queued allocates) that have been satisfied by “contention winner” sessions belonging to this mode group.
		Reset characteristic: reset to zero
Peak contention losers	A20E1HWM	is the peak number of “contention loser” sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM parameter of CEDA) as “contention winners” or “contention losers”, and their states are dynamically decided at bind time.
		Reset characteristic: reset to current value
Peak contention winners	A20E2HWM	is the peak number of “contention winner” sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as “contention winners” or “contention losers”, and their states are dynamically decided at bind time.
		Reset characteristic: reset to current value
The following three fields are incremented when an allocate is issued against a specific mode group. If a generic allocate request is made, the equivalent system entry statistics ***only*** are incremented.		
Peak outstanding allocates	A20ESTAM	is the peak number of allocation requests that were queued for this mode group.
		Reset characteristic: reset to current value

DFHSTUP name	Field name	Description
Total specific allocate requests	A20ESTAS	is the number of specific allocate requests against this mode group. Reset characteristic: reset to zero
Total specific allocates satisfied	A20ESTAP	is the number of specific allocates satisfied by this mode group. Reset characteristic: reset to zero
Total generic allocates satisfied	A20ESTAG	is the number of generic allocates satisfied from this mode group. The allocates are made for APPC without the mode group being specified. Reset characteristic: reset to zero
The following three fields are incremented when an allocate is issued against a specific mode group. If a generic allocate request is made, the equivalent system entry statistics ***only*** are incremented.		
Queued allocates	A20ESTAQ	is the current number of queued specific allocate requests against this mode group. An allocate is queued due to a session in this mode group not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use. Reset characteristic: not reset
Failed link allocates	A20ESTAF	is the number of specific allocate requests that failed due to the connection being released, out of service, or with a closed mode group. Reset characteristic: reset to zero
Failed allocates due to sessions in use	A20ESTAO	is the number of specific allocate requests that failed due to a session not being currently available for use in this mode group. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code. Reset characteristic: reset to zero
Number of XZIQUE allocate queue purges	A20EQPCT	is the total number of allocate queue purges that have occurred at XZIQUE request for this mode entry. Reset characteristic: reset to zero
Number of XZIQUE allocates purged	A20EZQPC	is the total number of allocates purged due to XZIQUE requesting that queues should be purged (A20EQPCT) for this mode entry. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation. Reset characteristic: reset to zero
Total bids sent	A20ESBID	is the number of bids that were sent on the sessions defined to this mode group. A bid is sent on an APPC “contention loser” session when there are no “contention winner” sessions available to allocate. Reset characteristic: reset to zero
Current bids in progress	A20EBID	is the number of bids that are in progress on the sessions defined to this mode group. A bid is sent on an APPC “contention loser” session when there are no “contention winner” sessions available to allocate. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Peak bids in progress	A20EBHWM	is the peak number of bids that were in progress at any one time, on the sessions defined to this mode group. A bid is sent on an APPC “contention loser” session when there are no “contention winner” sessions available to allocate. Reset characteristic: reset to current value

ISC/IRC mode entry: Summary resource statistics

Summary statistics are not available online.

These statistics are collected only if you have an APPC connection defined in your CICS region, and they are then produced for each mode group defined in that connection.

DFHSTUP name	Description
Connection name	is the name of the APPC connection/system that owns this mode entry. It corresponds to the system entry in the terminal definition.
Mode name	is the mode group name related to the intersystem connection name above (A20SYSN). It corresponds to the modename in the sessions definition.
ATIs satisfied by contention losers	is the total number of ATI requests (queued allocates) that have been satisfied by “contention loser” sessions belonging to this mode group.
ATIs satisfied by contention winners	is the total number of ATI requests (queued allocates) that have been satisfied by “contention winner” sessions belonging to this mode group.
Peak contention losers	is the peak number of “contention loser” sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as “contention winners” or “contention losers”, and their states are dynamically decided at bind time.
Peak contention winners	is the peak number of “contention winner” sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as “contention winners” or “contention losers”, and their states are dynamically decided at bind time.
The next three fields only contain allocates against specific mode groups. Generic allocate requests are contained in the equivalent system entry statistics.	
Peak outstanding allocates	is the peak number of allocation requests that were queued for this mode group.
Total specific allocate requests	is the total number of specific allocate requests against this mode group.
Total specific allocates satisfied	is the total number of specific allocates satisfied by this mode group.
Total generic allocates satisfied	is the total number of generic allocates satisfied from this mode group.
The next three fields only contain allocates against specific mode groups. Generic allocate requests are contained in the equivalent system entry statistics.	
Average number of queued allocates	is the average number of queued specific allocate requests against this mode group. An allocate is queued due to a session in this mode group not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use.
Failed link allocates	is the total number of specific allocate requests that failed due to the connection being released, out of service, or with a closed mode group.
Failed allocates due to sessions in use	is the total number of specific allocate requests that failed due to a session not being currently available for use in this mode group. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code.
Number of XZIQUE allocate queue purges	is the total number of allocate queue purges that have occurred at XZIQUE request for this mode entry.

DFHSTUP name	Description
Number of XZIQUE allocates purged	is the total number of allocates purged due to XZIQUE requesting that queues should be purged (A20EQPCT) for this mode entry. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation.
Total bids sent	is the total number of bids that were sent on the sessions defined to this mode group. A bid is sent on an APPC “contention loser” session when there are no “contention winner” sessions available to allocate.
Average bids in progress	is the average number of bids in progress.
Peak bids in progress	is the peak number of bids that were in progress at any one time, on the sessions defined to this mode group. A bid is sent on an APPC “contention loser” session when there are no “contention winner” sessions available to allocate.

ISC/IRC attach time entries

The ISC/IRC attach time statistics of the DFHSTUP listing is for a CICS system using intersystem communication or interregion communication. It provides summary statistics for the number of times that the entries on the PV 'signed on from' list are either reused or timed out. Using this data you can adjust the USRDELAY, and the PVDELAY system initialization parameters.

ISC/IRC attach time: Resource statistics

These statistics are collected if you have either an LU6.2 connection or IRC defined in your CICS region, and they are then produced globally, one per system. These statistics cannot be accessed online using the EXEC CICS COLLECT STATISTICS command; they are only produced for offline processing (written to SMF).

These statistics are mapped by the DFHA21DS DSECT.

DFHSTUP name	Field name	Description
Persistent Verification refresh time	A21_SIT_LUIT_TIME	is the time in minutes set by the PVDELAY parameter of the SIT. It specifies how long entries are allowed to remain unused in the PV 'signed on from' list of a remote system. The range is from zero through 10080 minutes (seven days) and the default is 30 minutes. If a value of zero is specified, then entries are deleted immediately after use. Reset characteristic: not reset
ISC Persistent Verification Activity		
Entries reused	A21_LUIT_TOTAL_REUSES	refers to the number of entries in the PV 'signed on from' list of a remote system that were reused without reference to an external security manager (ESM), such as RACF. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Entries timed out	A21_LUIT_TOTAL_TIMEOUT	refers to the number of entries in the PV 'signed on from' list of a remote system that were timed out. Reset characteristic: reset to zero
Average reuse time between entries	A21_LUIT_AV_REUSE_TIME	refers to the average time that has elapsed between each reuse of an entry in the PV 'signed on from' list of a remote system. Reset characteristic: reset to zero

ISC/IRC attach time: Summary resource statistics

Summary statistics are not available online.

These statistics are collected only if you have either an LU6.2 connection or IRC defined in your CICS region, and they are then produced globally, one per system.

DFHSTUP name	Description
Persistent verification refresh time	is the time in minutes set by the PVDELAY parameter of the SIT. It specifies how long entries are allowed to remain unused in the PV 'signed on from' list of a remote system.
refers to the average time that has elapsed between each reuse of a userid in the remote system.	

Journal statistics

Each journal employs two buffers. CICS uses one buffer for output while receiving records from transactions in the other buffer.

Journal control: Resource statistics

The information in this section is produced for every journal defined in the CICS region.

In addition to the number of records written, the journal control statistics indicates the frequency of the buffer full situation. This means that the receiving buffer has filled before I/O on the alternate buffer has had time to complete. The BUFSIZE operand for the journal should be increased to reduce this problem. These statistics are available online, and are mapped by the DFHA13DS DSECT.

DFHSTUP name	Field name	Description
Journal ID	A13JFID	is the identifier of the journal as specified in the JFILEID operand of DFHJCT. CICS itself use the first journal (JFILEID=01), called the system log. CICS users are allowed to have 98 other journals numbered 2 through 99, and can also write data on the system log. Journal identifiers appear in the same sequence as defined in the JCT. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Journal Type	A13JTYPE	is the journal type as defining in the TYPE=ENTRY JTYPE of the DFHJCT macro. JTYPE defines the type of journal being used. It can be one of the following: tapes, disks, or SMF. See the *CICS/ESA Resource Definition Guide* for the definition of JTYPE. Reset characteristic: not reset
Records Written	A13LRC	is the number of records written to the journal. Reset characteristic: reset to zero
Blocks Written	A13PBC	is the number of physical I/Os written to the journal. Reset characteristic: reset to zero
Buffer Full	A13BFC	is the number of times the receiving buffer filled before I/O on the alternate buffer has had a chance to complete. Increase the BUFSIZE value. Reset characteristic: reset to zero
Ave. O/P Blk size	A13ABS	is an approximate average of the output block size, expressed in bytes. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report. Reset characteristic: reset to zero
Last Vol Written	A13LVW	is the volume identifier of the most recently written volume (this is not applicable to disk journals or unlabeled tapes). **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report. Reset characteristic: not reset
Tapes Opened	A13VOOC	is the quantity of tapes that were opened for use. Reset characteristic: reset to zero
Tapes Left	A13STL	is, for journals on labeled tapes, the number of tapes in the series belonging to this journal that as yet bear no part of the journal. Reset characteristic: not reset
The following three statistics only have any meaning when automatic archiving has been specified in the JCT entry; the values are initialized to zero at CICS startup.		
Waits on Archive	A13WAC	is the number of times CICS has had to wait for this journal because the archive job has not completed at the time it was needed. Reset characteristic: reset to zero
Archives Submit.	A13ASUB	is the number of times an archive job has been sent to the JES queue for this journal. Reset characteristic: reset to zero
Data sets Opened	A13JDO	is the number of times an OPEN has been issued for this job. Reset characteristic: reset to zero

Journal control: Summary resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Journal Id	is the identifier of the journal as specified in the JCT JFILEID operand. CICS itself uses the first journal (JFILEID=01), called the system log. CICS users are allowed to have 98 other journals numbered 2 through 99, and can also write data on the system log. Journal identifiers appear in the same sequence as defined in the JCT.
Journal Type	indicates which volume the journal has been written to (DISK1 or DISK2) or whether the journal was written to an SMF data set.
Records written	is the total number of records written to the journal.
Blocks Written	is the total number of physical I/Os written to the journal.
Buffer Full	is the number of times the receiving buffer filled before I/O on the alternate buffer has had a chance to complete. Increase the BUFSIZE value.
Ave. O/P Blk size	is an approximate average of the output block size, expressed in bytes. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report.
Tapes Opened	is the total number of tapes that were opened for use.
The following three statistics only have any meaning when automatic archiving has been specified in the JCT entry; the values are initialized to zero at CICS startup.	
Waits on Archive	is the total number of times CICS has had to wait for this journal because the archive job has not completed at the time it was needed.
Archives Submit.	is the total number of times an archive job has been sent to the JES queue for this journal.
Data sets Opened	is the total number of times an OPEN has been issued for this job.

Loader statistics

The loader maintains global statistics to assist the user in tuning and accounting.

Loader domain: global statistics

These statistics fields contain the global data collected by the loader.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHLDGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

DFHSTUP name	Field name	Description
Library load requests	LDGLLR	is the number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure. Reset characteristic: reset to zero
Total loading time	LDGLLT	is the time taken for the number of library loads indicated by LDGLLR. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains a 4-byte field which expresses the time in 16-microsecond units. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Average loading time		is the average time to load a program. This value is calculated offline by DFHSTUP and hence is not available to online users. DFHSTUP expresses this time as *hours:minutes:seconds.decimals*. Reset characteristic: none
Program uses	LDGPUSES	is the number of uses of any program by the CICS system. Reset characteristic: not reset
Waiting requests	LDGWLR	is the number of loader domain requests that ***are currently*** forced to suspend due to the loader domain currently performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress. Reset characteristic: not reset
Requests that waited	LDGWTDLR	is the number of loader domain requests that ***were*** forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress. This figure is the total number of tasks that have waited, and does not include those that are currently waiting (LDGWLR). Reset characteristic: reset to zero
Peak waiting Loader requests	LDGWLRHW	is the maximum number of tasks suspended at one time. Reset characteristic: reset to current value (LDGWLR)
Times at peak	LDGHWMT	is the number of times the high watermark level indicated by LDGWLRHW was reached. This, along with the fields; LDGWTDLR and LDGWLRHW, is an indication of the level of contention for loader resource. Reset characteristic: reset to 1
Total waiting time	LDGTTW	is the suspended time for the number of tasks indicated by LDGWTDLR. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains a 4-byte field which expresses the time in 16-microsecond units. Reset characteristic: reset to zero
Times DFHRPL re-opened	LDGDREBS	is the number of times the loader received an end-of-extent condition during a LOAD and successfully closed and re-opened the DFHRPL library and retried the LOAD. Reset characteristic: reset to zero

CDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of CDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
ECDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of ECDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
SDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of SDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
ESDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of ESDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
RDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of RDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
ERDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of ERDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset

Loader domain: Summary global statistics

Summary statistics are not available online.

These statistics fields contain the summary global data for the loader.

DFHSTUP name	Description
Library load requests	is the total number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure.
Total loading time	is the total time taken for the number of library loads indicated by LDGLLR. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Average loading time	is the average time to load a program from the DFHRPL library concatenation into CICS managed storage. This value is expressed as *minutes:seconds.decimals*.
Program uses	is the total number of uses of any program by the CICS system.

DFHSTUP name	Description
Requests that waited	is the total number of loader domain requests that were forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress.
Peak waiting Loader requests	is the peak number of tasks suspended at one time.
Times at peak	is the total number of times the peak level indicated by the previous statistic was reached. This, along with the previous 2 values, is an indication of the level of contention for loader resource.
Total waiting time	is the total suspended time for the number of tasks indicated by the "Requests that waited" statistic. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Times DFHRPL re-opened	is the total number of times the loader received an end-of-extent condition during a LOAD and successfully closed and re-opened the DFHRPL library and retried the LOAD.
CDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ECDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.

DFHSTUP name	Description
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
SDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ESDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ESDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.

DFHSTUP name	Description
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
RDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ERDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days–hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.

DFHSTUP name	Description
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.

LSRPOOL statistics

CICS supports the use of up to eight LSRpools, and produces two sets of statistics for LSRpool activity.

LSRpool: resource statistics for each LSR pool

The following information describes the size and characteristics of the pool, and shows the data collected for the use of strings and buffers.

These statistics are available online, and are mapped by the DFHA08DS DSECT.

DFHSTUP name	Field name	Description
Pool Number	A08SRPID	is the identifying number of the pool. This value may be in the range 1 through 8.
		Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A08FLAGS	is a flag set to value X'80' if separate data and index pools are used, or set to value X'00' if data and index buffers share the same pool.
		Reset characteristic: not reset
Time Created	A08LKCTD	is the time when this LSR pool was created. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* in local time.
		Reset characteristic: not reset
Time Deleted	A08LKDTD	is the local time (STCK) when this LSR pool was deleted. This field is printed only if the pool has been deleted (that is, if all the files using the pool have been closed). If no value is set, the DSECT field contains the packed hexadecimal value X'00000000 00000000'
		This field is only printed for unsolicited statistics when the pool is deleted.
		The process of deleting an LSR pool results in the output of unsolicited statistics for the pool. Information for the deleted pool is not printed in subsequent statistics output. For this reason, the "time pool deleted" field is normally printed only in this unsolicited statistics output.
		Reset characteristic: not reset
NOT IN DFHSTUP REPORT	A08GBKCD	is the time when this LSR pool was created. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* in GMT.
		Reset characteristic: not reset

DFHSTUP name	Field name	Description
NOT IN DFHSTUP REPORT	A08GBKDD	is the time when this LSR pool was deleted expressed in GMT. This field is printed only if the pool has been deleted (that is, if all the files using the pool have been closed). If no value is set, the DSECT field contains the packed hexadecimal value X'00000000 00000000' This field is only printed for unsolicited statistics when the pool is deleted. The process of deleting an LSR pool results in the output of unsolicited statistics for the pool. Information for the deleted pool is not printed in subsequent statistics output. For this reason, the "time pool deleted" field is normally printed only in this unsolicited statistics output. Reset characteristic: not reset
Maximum key length	A08BKKYL	is the length of the largest key of a VSAM data set which may use the LSR pool. The value is obtained from one of: • The MAXKEYLENGTH option of the DEFINE LSRPOOL command in resource definition online, if it has been coded • A CICS calculation at the time the LSR pool is built. Reset characteristic: not reset
Total number of strings	A08BKSTN	is the value obtained from one of: • The STRINGS option of the DEFINE LSR command in resource definition online, if it has been coded • A CICS calculation at the time the LSR pool is built. Reset characteristic: not reset
Peak requests that waited for string	A08BKHSW	is the highest number of requests that were queued at one time because all the strings in the pool were in use. Reset characteristic: reset to current value
Total requests that waited for string	A08BKTSW	is the number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSR pool string resources. Reset characteristic: reset to zero
Peak concurrently active strings	A08BKHAS	is the maximum number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently lower than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need. Reset characteristic: reset to current value

Note that if separate data and index pools are not being used, all the statistics for the totals are obtained from the A08TO*xxx*_DATA variables, the index totals being unused.

LSRpool: data buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08TOBFN_DATA	is the number of data buffers used by the pool. Reset characteristic: not reset
Lookasides	A08TOBFF_DATA	is the number of successful lookasides to data buffers for the pool. Reset characteristic: not reset
Reads	A08TOFRD_DATA	is the number of read I/Os to the data buffers for the pool. Reset characteristic: not reset
User writes	A08TOUIW_DATA	is the number of user-initiated buffer WRITEs from data buffers for the pool. Reset characteristic: not reset
Non-user writes	A08TONUW_DATA	is the number of non-user-initiated buffer WRITEs from data buffers for the pool. Reset characteristic: not reset

Hiperspace: data buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08TOHBN_DATA	is the number of Hiperspace data buffers specified for the pool Reset characteristic: not reset
Hiperspace reads	A08TOCRS_DATA	is the number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers. Reset characteristic: not reset
Hiperspace writes	A08TOWRS_DATA	is the number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers. Reset characteristic: not reset
Hiperspace failed reads	A08TOCRF_DATA	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Reset characteristic: not reset
Hiperspace failed writes	A08TOCWF_DATA	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. Reset characteristic: not reset

LSRpool: index buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08TOBFN_INDX	is the number of index buffers used by the pool. Reset characteristic: not reset
Lookasides	A08TOBFF_INDX	is the number of successful lookasides to index buffers for the pool. Reset characteristic: not reset
Reads	A08TOFRD_INDX	is the number of read I/Os to the index buffers for the pool. Reset characteristic: not reset
User writes	A08TOUIW_INDX	is the number of user-initiated buffer WRITEs from index buffers for the pool. Reset characteristic: not reset
Non-user writes	A08TONUW_INDX	is the number of non-user-initiated buffer WRITEs from index buffers for the pool. Reset characteristic: not reset

Hiperspace index buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08TOHBN_INDX	is the number of Hiperspace index buffers specified for the pool Reset characteristic: not reset
Hiperspace reads	A08TOCRS_INDX	is the number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers. Reset characteristic: not reset
Hiperspace writes	A08TOWRS_INDX	is the number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers. Reset characteristic: not reset
Hiperspace failed reads	A08TOCRF_INDX	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Reset characteristic: not reset
Hiperspace failed writes	A08TOCWF_INDX	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. Reset characteristic: not reset

OR The following group of statistics fields describes the characteristics and usage of the different buffer sizes available for use by the pool. These statistics are available online, and are mapped by the A08BSSDS DSECT defined in the DFHA08DS DSECT. This DSECT is repeated for each of the 11 CISIZEs available.

LSRpool: Buffer statistics

DFHSTUP name	Field name	Description
Buffer Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built buffers to use. Reset characteristic: not reset
Number	A08BKBFN	lists the number of buffers of each size available to CICS: Reset characteristic: not reset
Lookasides	A08BKBFF	is the number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances. Reset characteristic: not reset
Reads	A08BKFRD	is the number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances. Reset characteristic: not reset
User writes	A08BKUIW	is the number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Non-user writes	A08BKNUW	is the number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances. Reset characteristic: not reset

Hiperspace buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08BKHBN	is the number of Hiperspace buffers specified for the pool. Reset characteristic: not reset
Hiperspace reads	A08BKCRS	is the number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers. Reset characteristic: not reset
Hiperspace writes	A08BKCWS	is the number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Reset characteristic: not reset
Hiperspace failed reads	A08BKCRF	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Reset characteristic: not reset
Hiperspace failed writes	A08BKCWF	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. Reset characteristic: not reset

These Hiperspace statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.

LSRpool: Summary resource statistics for each LSR pool

Summary statistics are not available online.

DFHSTUP name	Description
Total number of pools built	is the total number of LSRPOOLS that were built during the entire CICS run.
Peak requests that waited for string	is the highest number of requests that were queued at one time because all the strings in the pool were in use.
Total requests that waited for string	is the total number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSR pool string resources.

DFHSTUP name	Description
Peak concurrently active strings	is the peak number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently lower than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need.

LSRpool: Summary data buffer statistics

Summary statistics are not available online.

The group of statistics fields below summarizes the usage of each of the eight LSRPOOLS during the entire CICS run.

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	is the total number of successful lookasides to data buffers for the pool.
Reads	is the total number of read I/Os to the data buffers for the pool.
User writes	is the total number of user-initiated buffer WRITEs from data buffers for the pool.
Non-user writes	is the total number of non-user-initiated buffer WRITEs from data buffers for the pool.

Summary Hiperspace data buffer statistics

Summary statistics are not available online.

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of Hiperspace data buffers specified for the pool.
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

LSRpool: summary index buffer statistics

Summary statistics are not available online.

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	is the total number of successful lookasides to index buffers for the pool.
Reads	is the total number of read I/Os to the index buffers for the pool.
User writes	is the total number of user-initiated buffer WRITEs from index buffers for the pool.
Non-user writes	is the total number of non-user-initiated buffer WRITEs from index buffers for the pool.

LSRpool: Summary Hiperspace index buffer statistics

Summary statistics are not available online.

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers.
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

OR If LSRpool buffers are shared, the statistics that follow refer to those shared data and index buffers.

LSRpool: Summary resource statistics for each LSR pool

Summary statistics are not available online.

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	is the total number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.
Reads	is the total number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.
User writes	is the total number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.
Non-user writes	is the total number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.

LSRpool: summary Hiperspace buffer statistics

Summary statistics are not available online.

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers.
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. The above Hiperspace statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.

The following information describes the buffer usage for each file that was specified
to use the LSRpool at the time the statistics were collected.

When a file using an LSRPOOL is closed, an unsolicited statistics record is written
to SMF containing the LSRPOOL file statistics for that file. Therefore a complete
pitcure of the usage can only be obtained by running a SUMMARY report of the
statistics utility program, DFHSTUP, which aggregates both the USS and interval
statistics. These statistics are not available online.

LSRpool Files: Resource statistics for each file specified

to use the pool

DFHSTUP name	Field name	Description
Pool Number	A09SRPID	is the LSR pool number, in the range 1 through 8, associated with this file. Reset characteristic: not reset
File Name	A09DSID	is the CICS file identifier you specified through resource definition online. Reset characteristic: not reset
Data Buff Size	A09DBN	is the buffer size used for the file's data records. This value is one of the eleven possible VSAM buffer sizes ranging from 512 bytes to 32 KB. The value is zero if the file has not been opened yet. # **APAR PN91596** # MJO 23/1/97 # This field is not totalled in the DFHSTUP report. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Index Buff Size	A09IBN	is the buffer size used for the file's index records. This is printed, even if the file has subsequently been dynamically allocated to a VSAM ESDS or RRDS. The values this field may take are the same as for the data buffer size statistic. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report. Reset characteristic: not reset
Total Buff Waits	A09TBW	is the number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSR pool were in use. Reset characteristic: reset to zero
Peak Buff Waits	A09HBW	is the peak number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSR pool were in use. If the data sets are waiting for buffers you should examine the numbers of buffers defined for the data and index buffer sizes used by the data set. The buffer size used by VSAM depends on the control interval size in the VSAM definition of the data set. If no buffer size exists for the specified control interval size, the next largest buffer size available is used. Reset characteristic: reset to current value

LSRpool files: Summary resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Pool Number	is the LSR pool number, in the range 1 through 8, associated with this file.
File Name	is the CICS file identifier you specified through resource definition online.
Data Buff Size	is the last non-zero value encountered for the buffer size used for the file's data records. This value is one of the eleven possible VSAM buffer sizes ranging from 512 bytes to 32 KB. The value is zero if the file has not been opened yet. The last non-zero value is produced only if it has been opened. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report.
Index Buff Size	is the last non-zero value encountered for the buffer size used for the file's index records. This is printed, even if the file has subsequently been dynamically allocated to a VSAM ESDS or RRDS. The values this field may take are the same as for the data buffer size statistic. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report.
Total Buff Waits	is the total number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSR pool were in use.

DFHSTUP name	Description
Peak Buff Waits	is the peak number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSR pool were in use. If the data sets are waiting for buffers you should examine the numbers of buffers defined for the data and index buffer sizes used by the data set. The buffer size used by VSAM depends on the control interval size in the VSAM definition of the data set. If no buffer size exists for the specified control interval size, the next largest buffer size available is used.

Monitoring statistics

Monitoring data is made up of a combination of performance class data, exception class data, and SYSEVENT data.

Monitoring domain: Global statistics

These statistics fields are collected from the monitoring domain. They are available online, and are mapped by the DFHMNGDS DSECT.

DFHSTUP name	Field name	Description
Exception records	MNGER	is the number of exception records written to SMF. Reset characteristic: reset to zero
Exception records suppressed	MNGERS	is the number of exception records suppressed by the global user exit (XMNOUT). Reset characteristic: reset to zero
Performance records	MNGPR	is the number of performance records scheduled for output to SMF. Because the monitoring domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered. Reset characteristic: reset to zero
Performance records suppressed	MNGPRS	is the number of performance records suppressed by the global user exit (XMNOUT). Reset characteristic: reset to zero
SMF records	MNGSMFR	is the number of SMF records written to the SMF data set. CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing. Reset characteristic: reset to zero
SMF errors	MNGSMFE	is the number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive. Reset characteristic: reset to zero
Sysevent records	MNGSYSER	is the number of SYSEVENT notification records written to the MVS SRM (for later processing by RMF). A sysevent record is written at the completion of a transaction, or at suspend of a semi-permanent mirror (MRO only). Reset characteristic: reset to zero
Sysevent errors	MNGSYSEE	is the number of non-OK responses from the request to write a record to the MVS SRM (for later processing by RMF). This count is incremented when a SYSEVENT write fails for any reason, for example, when RMF in inactive. Reset characteristic: reset to zero

Monitoring domain: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Exception records	is the total number of exception records written to SMF.
Exception records suppressed	is the total number of exception records suppressed by the global user exit (XMNOUT).
Performance records	is the total number of performance records scheduled for output to SMF. Because the monitor domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered.
Performance records suppressed	is the total number of performance records suppressed by the global user exit (XMNOUT).
SMF records	is the total number of SMF records written to the SMF data set. CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing.
SMF errors	is the total number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive.
Sysevent records	is the total number of SYSEVENT notification records written to the MVS SRM (for later processing by RMF). A sysevent record is written at the completion of a transaction, or at suspend of a semi-permanent mirror (MRO only).
Sysevent errors	is the total number of non-OK responses from the request to write a record to the MVS SRM (for later processing by RMF). This count is incremented when a SYSEVENT write fails for any reason, for example, when RMF in inactive.

Program statistics

The program statistics assist the user in tuning and accounting.

Programs: Resource statistics

These statistics fields contain the resource data collected by the loader for each program. They are available online, and are mapped by the DFHLDRDS DSECT.

DFHSTUP name	Field name	Description
Program name	LDRPNAME	is the name of the program. Reset characteristic: not reset
Times used	LDRTU	is the number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue an MVS LOAD. Reset characteristic: reset to zero
Fetch count	LDRFC	is the number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	LDRFT	is the time taken to perform all fetches. The DSECT field contains a four-byte value that expresses the time in 16-microsecond units. Reset characteristic: reset to zero
Average fetch time	Calculated by DFHSTUP	is the average time taken to perform a fetch of the program. The DFHSTUP report expresses this time as *minutes:seconds.decimals*. Reset characteristic: reset to zero
RPL offset	LDRRPLO	is the offset into the DFHRPL DD concatenation of the library from which the program was last loaded or is loaded when next required non-LPA resident modules only. **Note:** The offset values begin with zero for the first partitioned data set in the concatenation and thus this field may not be used to deduce whether a copy of the program is available to the loader domain. # **APAR PN91596** # MJO 23/1/97 # This field is not totalled in the DFHSTUP report. Reset characteristic: not reset
NEWCOPY count	LDRTN	is the number of times a NEWCOPY has been requested against this program. Reset characteristic: reset to zero
Program size	LDRPSIZE	is the size of the program in bytes, if known (otherwise zero). # **APAR PN91596** # MJO 23/1/97 # This field is not totalled in the DFHSTUP report. Reset characteristic: not reset
Times removed	LDRRPC	is the number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Location	LDRLOCN	is the location of the current storage resident instance of the program, if any. It has one of the following values: **DFHSTUP value:** NONE **DSECT value:** LDRNOCO (X'00') **Meaning:** No current copy **DFHSTUP value:** CDSA **DSECT value:** LDRCDCO (X'01') **Meaning:** Current copy in the CDSA **DFHSTUP value:** SDSA **DSECT value:** LDRSDCO (X'08') **Meaning:** Current copy in the SDSA **DFHSTUP value:** LPA **DSECT value:** LDRLPACO (X'03') **Meaning:** Current copy in the LPA **DFHSTUP value:** ECDSA **DSECT value:** LDRECDCO (X'04') **Meaning:** Current copy in the ECDSA **DFHSTUP value:** ESDSA **DSECT value:** LDRESDCO (X'09') **Meaning:** Current copy in the ESDSA **DFHSTUP value:** ERDSA **DSECT value:** LDRERDCO (X'06') **Meaning:** Current copy in the ERDSA **DFHSTUP value:** RDSA **DSECT value:** LDRRDCO (X'0A') **Meaning:** Current copy in the RDSA Reset characteristic: not reset

Programs: Summary resource statistics

Summary statistics are not available online.

These statistics fields contain the summary resource data statistics for the loader for each program.

DFHSTUP name	Description
Program name	is the name of the program.
Times used	is the total number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue MVS LOAD requests to obtain access to usable instances of this program.
Fetch count	is the total number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage.
Average fetch time	is the average time taken to perform a fetch of the program. The DFHSTUP report expresses this time as *minutes:seconds.decimals*.
NEWCOPY count	is the total number of times a NEWCOPY has been requested against this program.
Times removed	is the total number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism.

Program autoinstall statistics

Program autoinstall: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHPGGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

DFHSTUP name	Field name	Description
Program autoinstall attempts	PGGATT	is the number of times that a program autoinstall was attempted.
		Reset characteristic: reset to zero
Rejected by autoinstall exit	PGGREJ	is the number of times that a program autoinstall request was rejected by the program autoinstall URM program.
		Reset characteristic: reset to zero
Failed autoinstall attempts	PGGFAIL	is the number of times that a program autoinstall failed due to a number of reasons other than rejects (as counted by PGGREJ). For example the autoinstall URM program did not provide valid attributes; the model name specified by the URM was not defined; the exit tried to recurse, and disabled the URM.
		Reset characteristic: reset to zero

Program autoinstall: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Program autoinstall attempts	is the number of times that a program was autoinstalled.
Rejected by autoinstall exit	is the number of times that a program is rejected by the autoinstall exit.
Failed autoinstall attempts	is the number of times that a program failed to autoinstall.

Statistics domain statistics

Statistics domain: Global statistics

These statistics are available online, and are mapped by the DFHSTGDS DSECT.

DFHSTUP name	Field name	Description
Interval collections so far	STGNC	is the number of interval collections made during the CICS run, or from one end-of-day to the following end-of-day. Reset characteristic: This field is reset to zero only at every end-of-day collection.
SMF writes	STGSMFW	is the number of SMF writes since the last reset time. This figure includes records written for all types of statistics collections. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	STGLDW	is the length of data written to SMF during an interval, expressed as bytes. This figure includes length of data written during an interval for unsolicited, requested, and interval/end-of-day collections. Reset characteristic: reset to zero **Note:** This field contains the accumulated length of statistics records excluding the SMF headers.

Interval, end-of-day, and requested statistics all contain the same items.

Statistics domain: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Total number of interval collections	is the total number of interval collections made during the entire CICS run.
Total number of SMF writes	is the total number of SMF writes during the entire CICS run. This figure includes records written during an interval for unsolicited, requested, and interval/end-of-day collections.

Storage manager statistics

These statistics are produced to aid all aspects of storage management.

Storage manager statistics: Domain subpools

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHSMDDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

DFHSTUP name	Field name	Description
Subpool Name	SMDSPN	is the unique 8-character name of the domain subpool. The values of the domain subpool field are described in Appendix C, "MVS and CICS virtual storage" on page 493 . Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDETYPE	The assembler DSECT field name has the value X'01' or X'02', indicating whether all the elements in the subpool are fixed length or variable length. For further information about subpool elements, see Appendix C, "MVS and CICS virtual storage" on page 493 . Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDFLEN	is the length of each subpool element (applicable to FIXED length subpools only). For further information about subpool elements, see Appendix C, "MVS and CICS virtual storage" on page 493 . Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDELCHN	The assembler DSECT field name has the value X'01' or X'02', indicating whether or not SM maintains an element chain for the subpool with the addresses and lengths of each element. For further information about element chains, see Appendix C, "MVS and CICS virtual storage" on page 493 . Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDBNDRY	is the boundary on which each element is aligned. This is a power of 2 in the range 8 through 4096 bytes. For further information about boundaries, see Appendix C, "MVS and CICS virtual storage" on page 493 . Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDLOCN	The storage location of this domain subpool. The assembler DSECT field name has the following values: • SMDBELOW (X'01') below the 16MB line. • SMDABOVE (X'02') above the 16MB line. Reset characteristic: not reset
Location	SMDDSANAME	Name of the DSA that the domain subpool is allocated from. Values can be 'CDSA', 'SDSA', 'RDSA', 'ECDSA', 'ESDSA', and 'ERDSA'. Reset characteristic: not reset

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMDDSAINDEX	A unique identifier for the dynamic storage area that this subpool is allocated from. Values can be • SMDCDSA (X'01') indicating that the subpool storage is obtained from the CDSA. • SMDSDSA (X'03') indicating that the subpool storage is obtained from the UDSA. • SMDRDSA (X'04') indicating that the subpool storage is obtained from the UDSA. • SMDECDSA (X'05') indicating that the subpool storage is obtained from the ECDSA. • SMDESDSA (X'07') indicating that the subpool storage is obtained from the EUDSA. • SMDERDSA (X'08') indicating that the subpool storage is obtained from the ERDSA. Reset characteristic: not reset
Access	SMDACCESS	is the type of access of the subpool. It will be either CICS, USER, or READONLY. If storage protection not active, all storage areas will revert to CICS except those in the ERDSA. • SMDCICS (X'01') access is CICS key. • SMDUSER (X'02') access is USER key. • SMDREADONLY (X'03') is read-only protection. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDIFREE	is the size of the initial free area for the subpool (which may be zero), expressed in bytes. For further information about the initial free area, see Appendix C, “MVS and CICS virtual storage” on page 493 . Reset characteristic: not reset
Getmain Requests	SMDGMREQ	is the number of GETMAIN requests for the subpool. Reset characteristic: reset to zero
Freemain Requests	SMDFMREQ	is the number of FREEMAIN requests for the subpool. Reset characteristic: reset to zero
Current Elements	SMDCELEM	is the current number of storage elements in the subpool. Reset characteristic: not reset
Current Elem Stg	SMDCES	is the sum of the lengths of all the elements in the subpool, expressed in bytes. Reset characteristic: not reset
Current Page Stg	SMDCPS	is the space taken by all the pages allocated to the subpool, expressed in bytes. Reset characteristic: not reset
Peak Page Stg	SMDHWMPS	is the peak page storage allocated to support the storage requirements of this subpool. Reset characteristic: reset to current value

Summary domain subpools statistics

Summary statistics are not available online.

DFHSTUP name	Description
Subpool Name	is the unique 8-character name of the domain subpool. The values of the domain subpool field are described in Appendix C, “MVS and CICS virtual storage” on page 493 .
Location	is the indicator of the subpool location (CDSA, SDSA, RDSA, ECDSA, ESDSA, or ERDSA).
Access	is the type of access of the subpool. It will be either CICS, USER, or READONLY. If storage protection not active, all storage areas will revert to CICS except those in the ERDSA.
Getmain Requests	is the total number of GETMAIN requests for the subpool.
Freemain Requests	is the total number of FREEMAIN requests for the subpool.
Peak Elements	is the peak number of storage elements in the subpool.
Peak Elem Stg	is the peak amount of element storage in the subpool, expressed in bytes.
Peak Page Stg	is the peak amount of page storage in the subpool, expressed in bytes.

Storage manager: global statistics

These statistics ***can*** be accessed online using the EXEC CICS COLLECT STATISTICS command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

These statistics are collected for each pagepool. They are available online, and are mapped by the DFHSMSDS DSECT.

DFHSTUP name	Field name	Description
Storage protection	SMSSTGPROT	X'01' active
		X'00' not active
Transaction isolation	SMSTRANISO	X'01' active
		X'00' not active
Reentrant programs	SMSRENTPGM	X'01' protect. RDSA and ERDSA obtained from key 0 storage.
		X'00' no protect. RDSA and ERDSA obtained from key 8 storage.
Current DSA limit	SMSDSALIMIT	Current DSA limit value
Current DSA total	SMSDSATOTAL	Total amount of storage currently allocated to the DSAs below the line. This value may be smaller or larger than SMSDSALIMIT.
Peak DSA total	SMSHWMDSATOTAL	The total amount of storage allocated to the DSAs below the line. This value may be smaller or larger than SMSDSALIMIT.
Current EDSDA limit	SMSEDSALIMIT	Current EDSA limit
Current EDSA total	SMSEDSATOTAL	Total amount of storage currently allocated to the DSAs above the line. This value may be smaller or larger than EDSALIMIT.

DFHSTUP name	Field name	Description
Peak EDSA total	SMSHWMESDATOTAL	The total amount of storage allocated to the DSAs above the line. This value may be smaller or larger than SMSESDALIMIT.

Storage manager: subspace statistics

These statistics ***can*** be accessed online using the EXEC CICS COLLECT STATISTICS command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

These statistics are collected for each pagepool, and are mapped by the DFHSMSDS DSECT.

DFHSTUP name	Field name	Description
Current unique subspace users	SMSUSSCUR	Current number of unique subspace users. Number of tasks currently allocated a unique subspace.
Total unique subspace users	SMSUSSCUM	Total number of tasks that have been allocated a unique subspace.
Peak unique subspace users	SMSUSSHWM	The peak number of tasks concurrently allocated a unique subspace.
Current common subspace users	SMSCSSCUR	Number of tasks currently allocated to the common subspace
Total common subspace users	SMSCSSCUM	Total number of tasks allocated to the common subspace
Peak common subspace users	SMSCSSHWM	The peak number of tasks concurrently allocated to the common subspace.

Storage manager statistics: dynamic storage areas

These statistics ***can*** be accessed online using the EXEC CICS COLLECT STATISTICS command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

These statistics are collected for each pagepool. They are available online, and are mapped by the DFHSMSDS DSECT.

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMSNPAGP	is the number of pagepools in the CICS region. There are eight pagepools: the CDSA (CICS dynamic storage area), the UDSA (user dynamic storage area), the SDSA (shared dynamic storage area), the RDSA (read-only dynamic storage area), the ECDSA (extended CICS dynamic storage area), the ESDSA (extended shared dynamic storage area), the EUDSA (extended user dynamic storage area), and the ERDSA (extended read-only dynamic storage area). Reset characteristic: not reset
Note: The following fields are mapped by the SMSBODY DSECT within the DFHSMSDS DSECT. The SMSBODY DSECT is repeated for each pagepool in the CICS region (SMSNPAGP).		
Header in DFHSTUP report	SMSDSANAME	Name of the DSA that this record represents. Values can be 'CDSA', 'UDSA', 'SDSA', 'RDSA', 'ECDSA', 'EUDSA', 'ESDSA', and 'ERDSA'. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMSDSAINDEX	A unique identifier for the dynamic storage area that this subpool is allocated from. Values can be: • SMSCDSA (X'01') The page pool is the CDSA. • SMSUDSA (X'02') The page pool is the UDSA. • SMSSDSA (X'03') The page pool is the SDSA. • SMSRDSA (X'04') The page pool is the RDSA. • SMSECDSA (X'05') The page pool is the ECDSA. • SMSEUDSA (X'06') The page pool is the EUDSA. • SMSESDSA (X'07') The page pool is the ESDSA. • SMSERDSA (X'08') The page pool is the ERDSA. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMSLOCN	is the location of this pagepool. The assembler DSECT field name has the following values: • SMSBELOW (X'01') below the 16MB line. • SMSABOVE (X'02') above the 16MB line.
Current DSA Size	SMSDSASZ	is the current size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes. Reset characteristic: not reset

DFHSTUP name	Field name	Description
Peak DSA Size	SMSHWMDSASZ	is the peak size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes sinces that last time that statistics were recorded.
		Reset characteristic: not reset
Cushion Size	SMSCSIZE	is the size of the cushion, expressed in bytes. The cushion forms part of the CDSA, UDSA, ECDSA, EUDSA, or the ERDSA, and is the amount of storage below which CICS goes SOS.
		Reset characteristic: not reset
Free storage (inc. cushion)	SMSFSTG	is the amount of free storage in this pagepool, that is the number of free pages multiplied by the page size (4K), expressed in bytes.
		Reset characteristic: not reset
Percentage free storage		is the percentage of the storage that is free. This value is calculated offline by DFHSTUP and is, therefore, not accessible via the EXEC CICS COLLECT STATISTICS command.
		Reset characteristic: not reset
Peak free storage	SMSHWMFSTG	is the largest amount of storage that is free since the last time that statistics were recorded.
		Reset characteristic: not reset
Lowest free storage	SMSLWMFSTG	is the smallest amount of storage that is free since the last time that statistics were recorded.
		Reset characteristic: not reset
Largest free area	SMSLFA	is the length of the largest contiguous free area in the CDSA RDSA, SDSA, EDSA, UDSA, ECDSA, EUDSA, or ERDSA, expressed in bytes. To get an indication of the storage fragmentation in this pagepool, compare this value with “Free storage” (SMSFSTG) in the pagepool. If the ratio is large, then this pagepool is fragmented.
		Reset characteristic: not reset
Getmain Requests	SMSGMREQ	is the number of GETMAIN requests from the CDSA, RDSA, SDSA, EDSA, UDSA, or ECDSA, EUDSA, or ERDSA.
		Reset characteristic: reset to zero
Freemain Requests	SMSFMREQ	is the number of FREEMAIN requests from the CDSA, UDSA, ECDSA, EUDSA, RDSA, SDSA, EDSA, or ERDSA.
		Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	SMSASR	is the number of ADD_SUBPOOL requests to create a subpool (domain or task) from the CDSA, UDSA, ECDSA, EUDSA, ERDSA, RDSA, SDSA, or ESDSA.
		Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMSDSR	is the number of DELETE_SUBPOOL requests (domain or task) from the CDSA, UDSA, ECDSA, EUDSA, ERDSA, RDSA, SDSA, or ESDSA. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	SMSCSUBP	is the current number of subpools (domain and task) in the CDSA, UDSA, ECDSA, EUDSA, ERDSA, RDSA, SDSA, or ESDSA. Reset characteristic: not reset
Times no storage returned	SMSCRISS	is the number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Reset characteristic: reset to zero
Times request suspended	SMSUCSS	is the number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. Reset characteristic: reset to zero
Current suspended	SMSCSS	is the number of GETMAIN requests currently suspended for storage. Reset characteristic: not reset
Peak requests suspended	SMSHWMSS	is the peak number of GETMAIN requests suspended for storage. Reset characteristic: reset to current value
Purged while waiting	SMSPWWS	is the number of requests which were purged while suspended for storage. Reset characteristic: reset to zero
Times cushion released	SMSCREL	is the number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion Reset characteristic: reset to zero
Times went short on storage	SMSSOS	is the number of times CICS went SOS in this pagepool (CDSA, UDSA, ECDSA, EUDSA, RDSA, SDSA, ESDSA or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. Reset characteristic: reset to zero
Total time SOS	SMSTSOS	is the accumulated time that CICS has been SOS in this DSA. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Storage violations	SMSSV	is the number of storage violations recorded in the CDSA, UDSA, ECDSA, EUDSA, RDSA, SDSA, ESDSA, and the ERDSA.
Access	SMSACCESS	is the type of access of the page subpool. It will be either CICS, USER, or READONLY. If storage protection not active, all storage areas will revert to CICS except those in the ERDSA. • SMSCICS (X'01') access is CICS key. • SMSUSER (X'02') access is USER key. • SMSREADONLY (X'03') is read-only protection. Reset characteristic: not reset
Current extents	SMSEXTS	is the number of extents currently allocated to a specified dynamic storage area. Reset characteristic: not reset
Extents added	SMSEXTSA	is the number of extents added to a dynamic storage area since the last time statistics were recorded. Reset characteristic: not reset
Extents released	SMSEXTSR	is the number of extents which have been released from a dynamic storage area since the last time statistics were recorded. Reset characteristic: not reset

Storage manager: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Storage protection	is the total storage protection defined to the storage manager.
Transaction isolation	is the number of transactions associated with this terminal that can be isolated.
Reentrant programs	is the number of programs that have used the reentry facility.
Current DSA limit	this is the limit of CICS dynamic storage area that can be defined by the storage manager
Current DSA total	this is the number of CICS dynamic storage areas currently in use by the storage manager
Peak DSA total	is the highest number of CICS dynamic storage areas used by the storage manager since the last recorded statistics
Current EDSA limit	this is the number of extended dynamic storage areas currently defined by the storage manager
Current EDSA total	this is the number of extended dynamic storage areas currently in use by the storage manager
Peak EDSA total	is the highest number of extended dynamic storage area defined by the storage manager since the last recorded statistics

Storage manager: Summary subspace statistics

Summary statistics are not available online.

DFHSTUP name	Description	
Total unique subspace users	SMSUSSCUM	Total number of tasks that have been allocated a unique subspace.
Peak unique subspace users	SMSUSSHWM	The peak number of tasks concurrently allocated a unique subspace.
Total common subspace users	SMSCSSCUM	Total number of tasks allocated to the common subspace
Peak common subspace users	SMSCSSHWM	The peak number of tasks concurrently allocated to the common subspace.

Summary dynamic storage areas statistics

Summary statistics are not available online.

DFHSTUP name	Description
DSA size	is the total size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes.
Cushion size	is the size of the cushion, expressed in bytes. The cushion forms part of the DSA or the EDSA, and is the amount of storage below which CICS goes SOS.
Getmain requests	is the total number of GETMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA.
Freemain requests	is the total number of FREEMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA.
Times no storage returned	is the total number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE.
Times request suspended	is the total number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment.
Peak requests suspended	is the peak number of GETMAIN requests suspended for storage.
Purged while waiting	is the total number of requests which were purged while suspended for storage.
Times cushion released	is the total number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion
Times went short on storage	is the total number of times CICS went SOS in this pagepool (CDSA, UDSA, ECDSA, EUDSA, or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage.
Total time SOS	is the accumulated time that CICS has been SOS in this DSA. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value.
Storage violations	is the total number of storage violations recorded in the CDSA, UDSA, ECDSA, EUDSA, and the ERDSA.
Access	is the type of access of the page subpool. It will be either CICS, USER, or READONLY. If storage protection not active, all storage areas will revert to CICS except those in the ERDSA.

Storage manager statistics: Task subpools

These statistics cannot be accessed online using the EXEC CICS COLLECT STATISTICS command. They are produced only for offline processing (written to SMF).

These statistics are collected for each pagepool. They are mapped by the DFHSMTDS DSECT.

Although task subpools are dynamically created and deleted for each task in the system, these statistics are the sum of all task subpool figures for the task related pagepools (CDSA, UDSA, ECDSA, and EUDSA). If further granularity of task storage usage is required, use the performance class data of the CICS/ESA monitoring facility.

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMTNTASK	is the number of task subpools in the CICS region. Reset characteristic: not reset

Note: The following fields are mapped by the SMTBODY DSECT within the DFHSMTDS DSECT. The SMTBODY DSECT is repeated for each task subpool in the CICS region (SMTNTASK).

DSA Name	SMTDSANAME	Name of the dynamic storage area from which this task storage has been allocated. Values can be 'CDSA', 'UDSA', 'ECDSA',and 'EUDSA'. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMTDSAINDEX	A unique identifier for the dynamic storage area that these statistics refer to. Values can be: • SMTCDSA (X'01') indicating that the task storage is obtained from the CDSA • SMTUDSA (X'02') indicating that the task storage is obtained from the UDSA • SMTECDSA (X'05') indicating that the task storage is obtained from the ECDSA • SMTEUDSA (X'06') indicating that the task storage is obtained from the EUDSA Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMTLOCN	tells you whether the dynamic storage area is above or below the line. • SMTBELOW (X'01') below the 16MB line. • SMTABOVE (X'02') above the 16MB line. Reset characteristic: not reset
Access	SMTACCESS	is the type of access of the subpool. It will be either CICS or USER. • SMTCICS (X'01') access is CICS key. • SMTUSER (X'02') access is USER key. Reset characteristic: not reset
Getmain Requests	SMTGMREQ	is the number of task subpool GETMAIN requests from this dynamic storage area. Reset characteristic: reset to zero
Freemain Requests	SMTFMREQ	is the number of task subpool FREEMAIN requests from this dynamic storage area. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Current Elements	SMTCNE	is the number of elements in all the task subpools in this dynamic storage area.
		Reset characteristic: not reset
Current Elem Stg	SMTCES	is the sum of the storage occupied by all elements in task subpools within this dynamic storage area, expressed in bytes.
		Reset characteristic: not reset
Current Page Stg	SMTCPS	is the sum of the storage in all pages allocated to task subpools within this dynamic storage area, expressed in bytes.
		Reset characteristic: not reset
Peak Page Stg	SMTHWMPS	is the peak page storage allocated to support task storage activity in that dynamic storage area.
		Reset characteristic: reset to current value

Summary task subpools statistics

Summary statistics are not available online.

The following fields are mapped by the SMTBODY DSECT within the DFHSMTDS DSECT. The SMTBODY DSECT is repeated for each task subpool in the CICS region (SMTNTASK).

DFHSTUP name	Description
DSA Name	tells you whether the dynamic storage area is in the CDSA, UDSA, ECDSA, or EUDSA.
Access	is the type of access of the subpool. It will be either CICS, or USER.
Getmain Requests	is the total number of task subpool GETMAIN requests from this dynamic storage area.
Freemain Requests	is the total number of task subpool FREEMAIN requests from this dynamic storage area.
Peak Elements	is the peak number of elements in all the task subpools in this dynamic storage area.
Peak Elem Storage	is the peak amount of storage occupied by all elements in task subpools within this dynamic storage area, expressed in bytes.
Peak Page Storage	is the peak amount of storage in all pages allocated to task subpools within this dynamic storage area, expressed in bytes.

Table manager

Table manager: Global statistics

These statistics are available online, and are mapped by the DFHA16DS DSECT.

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A16NTAB	is the number of tables defined to the table manager. Reset characteristic: not reset
The following fields are mapped by the A16STATS DSECT, which is repeated for each table (A16NTAB).		
Table Name	A16TNAM	is the name of a CICS table supported by the table manager. Reset characteristic: not reset
Total Size of Table Manager Storage (bytes)	A16TSIZE	is the amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves. Reset characteristic: not reset

Table manager: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Table Name	is the name of a CICS table supported by the table manager.
Average Table Size (bytes)	is the average amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves.
Peak Table Size (bytes)	is the peak amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves.

Temporary storage

Temporary storage statistics are produced for the data that is written into a temporary storage queue.

This is the DFHSTUP listing for temporary storage statistics.

Temporary storage: Global statistics

These statistics are available online, and are mapped by the DFHA12DS DSECT.

DFHSTUP name	Field name	Description
Put/Putq main storage requests	A12STA5F	is the number of records that application programs wrote to main temporary storage.
		Reset characteristic: reset to zero
Get/Getq main storage requests	A12NMG	is the number of records that application programs obtained from main temporary storage.
		Reset characteristic: reset to zero
Peak storage for temp. storage (main)	A12STA6F	is the peak value, expressed in bytes, of the amount of virtual storage used for temporary storage records.
		Reset characteristic: reset to current value
Current storage for temp. storage (main)	A12STA6A	is the current value, expressed in bytes, of the amount of virtual storage used for temporary storage records.
		Reset characteristic: not reset
Put/Putq auxiliary storage requests	A12STA7F	is the number of records that application programs wrote to auxiliary temporary storage.
		Reset characteristic: reset to zero
Get/Getq auxiliary storage requests	A12NAG	is the number of records that application programs obtained from auxiliary temporary storage.
		Reset characteristic: reset to zero
Peak temporary storage names in use	A12QNUMH	is the peak number of temporary storage queue names in use at any one time.
		Reset characteristic: reset to current value
Current temporary storage names in use	A12QNUM	is the current number of temporary storage queue names in use.
		Reset characteristic: not reset
Number of entries in longest queue	A12QINH	is the peak number of items in any one queue.
		Reset characteristic: reset to zero
Queue extensions threshold	A12GIDNE	is the number of records that are held in a single temporary storage group identifier (TSGID). This value is controlled by the system initialization parameter, TSMGSET.
		Reset characteristic: not reset

Note: The next two items are indications of whether the value chosen for TSMGSET is working as planned.

Times queues created	A12STA3F	is the number of times that CICS created individual temporary storage queues.
		Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Queue extensions created	A12STA4F	is the number of times it was necessary to create a TSGID extension. The capacity of a single TSGID is determined by the TSMGSET option in the SIT, and indirectly controls how many additional extension blocks have to be built.
		Reset characteristic: reset to zero
Control interval size	A12CSZ	is the size of VSAM's unit of transmission between DASD and main storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set (for guidance information about this, see the *CICS/ESA Operations and Utilities Guide*). In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead.
		Reset characteristic: not reset
Available bytes per control interval	A12NAVB	is the number of bytes available for use in the TS data set control interval.
		Reset characteristic: not reset
Segments per control interval	A12SPCI	is the number of segments available in the TS control interval.
		Reset characteristic: not reset
Bytes per segment	A12BPSEG	is the number of bytes per segment of the TS data set.
		Reset characteristic: not reset
Writes more than control interval	A12STABF	is the number of writes of records whose length was greater than the control interval (CI) size.
		Reset characteristic: reset to zero
Longest auxiliary temporary storage record	A12LAR	is the size, expressed in bytes, of the longest record written to the temporary storage data set.
		Reset characteristic: not reset
Number of control intervals available	A12NCI	is the number of control intervals (CIs) available for auxiliary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination.
		Reset characteristic: not reset
Peak control intervals in use	A12NCIAH	is the peak number of CIs containing active data.
		Reset characteristic: reset to current value
Current control intervals in use	A12NCIA	is the current number of CIs containing active data.
		Reset characteristic: not reset
Times aux. storage exhausted	A12STA8F	is the number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced to abend.
		Reset characteristic: reset to zero
Number of temp. storage compressions	A12STA9F	is the number of times that the temporary storage buffers were compressed.
		Reset characteristic: reset to zero

Note: The following statistics are produced for buffer usage:

DFHSTUP name	Field name	Description
Temporary storage buffers	A12NBCA	is the number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides. The number of buffers allocated may exceed the number requested.
		Reset characteristic: not reset
Buffer waits	A12BWTN	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.
		Reset characteristic: reset to zero
Peak users waiting on buffer	A12BUWTH	is the peak number of requests queued because no buffers were available.
		Reset characteristic: reset to current value
Current users waiting on buffer	A12BUWT	is the current number of requests queued because no buffers were available.
		Reset characteristic: not reset
Buffer writes	A12TWTN	is the number of WRITEs to the temporary storage data set. This includes both WRITEs necessitated by recovery requirements (see next item) and WRITEs forced by the buffer being needed to accommodate another CI.
		Reset characteristic: reset to zero
Forced writes for recovery	A12TWTNR	is the subset of the total number of WRITEs caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation.
		Reset characteristic: reset to zero
Buffer reads	A12TRDN	is the number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
		Reset characteristic: reset to zero
Format writes	A12TWTNF	is the number of times a new CI was successfully written at the end of the data set to increase the amount of available space in the data set. A formatted write is attempted only if the current number of CIs available in the auxiliary data set have all been used.
		Reset characteristic: reset to zero
Note: The following statistics are produced for string usage:		
Temporary storage strings	A12NVCA	is the number of temporary storage strings specified in the TS= system initialization parameter or in the overrides. The number of strings allocated may exceed the number requested.
		Reset characteristic: not reset
Peak number of strings in use	A12NVCAH	is the peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number.
		Reset characteristic: reset to current value

DFHSTUP name	Field name	Description
Times string wait occurred	A12VWTN	is the number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the number of I/O requests, consider increasing the number of strings initially allocated. Reset characteristic: reset to zero
Peak number of users waiting on string	A12VUWTH	is the peak number of I/O requests that were queued at any one time because all strings were in use. Reset characteristic: reset to current value
Current users waiting on string	A12VUWT	is the current number of I/O requests that are queued because all strings are in use. Reset characteristic: not reset
I/O errors on TS data set	A12STAAF	is the number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause. Reset characteristic: reset to zero

Temporary storage: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Put/Putq main storage requests	is the total number of records that application programs wrote to main temporary storage.
Get/Getq main storage requests	is the total number of records that application programs obtained from main temporary storage.
Peak storage for temp. storage (main)	is the peak value, expressed in bytes, of the amount of virtual storage used for temporary storage records.
Put/Putq auxiliary storage requests	is the total number of records that application programs wrote to auxiliary temporary storage.
Get/Getq auxiliary storage requests	is the total number of records that application programs obtained from auxiliary temporary storage.
Peak temporary storage names in use	is the peak number of temporary storage queue names at any one time.
Number of entries in longest queue	is the peak number of items in any one queue, up to a maximum of 32KB.
Queue extensions threshold	is the total number of records that are held in a single temporary storage group identifier (TSGID). This value is controlled by the system initialization parameter, TSMGSET.

Note: The next two items are indications of whether the value chosen for TSMGSET is working as planned.

Times queues created	is the total number of times that CICS created individual temporary storage queues.
Queue extensions created	is the total number of times it was necessary to create a TSGID extension. The capacity of a single TSGID is determined by the TSMGSET option in the SIT, and indirectly controls how many additional extension blocks have to be built.

DFHSTUP name	Description
Control interval size	is the size of VSAM's unit of transmission between DASD and main storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set (for guidance information about this, see the *CICS/ESA Operations and Utilities Guide*). In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead.
Available bytes per control interval	is the number of bytes available for use in the last TS data set control interval.
Segments per control interval	is the number of segments in last TS data set control interval.
Bytes per segment	is the number of bytes per segment of the last TS data set control interval.
Writes more than control interval	is the total number of writes of records whose length was greater than the control interval (CI) size. If the reported value is large, increase the CI size. If the value is zero, consider reducing the CI size until a small value is reported.
Longest auxiliary temporary storage record	is the size, expressed in bytes, of the longest record written to the temporary storage data set.
Number of control intervals available	is the number of control intervals (CIs) available for auxiliary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination.
Peak control intervals available	is the peak number of CIs containing active data.
Times aux. storage exhausted	is the total number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced to abend. If this item appears in the statistics, increase the size of the temporary storage data set.
Number of temp. storage compressions	is the total number of times that temporary storage buffers were compressed.

Note: The following statistics are produced for buffer usage:

Temporary storage buffers	is the total number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides. The number of buffers allocated may exceed the number requested.
Buffer waits	is the total number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.
Peak users waiting on buffers	is the peak number of requests queued because no buffers were available.
Buffer writes	is the total number of WRITEs to the temporary storage data set. This includes both WRITEs necessitated by recovery requirements (see next item) and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing buffer allocation.
Forced writes for recovery	is the subset of the total number of WRITEs caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation.
Buffer reads	is the total number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
Format writes	is the total number of times a new CI was successfully written at the end of the data set to increase the amount of available space in the data set. A formatted write is attempted only if the current number of CIs available in the auxiliary data set have all been used.

Note: The following statistics are produced for string usage:

DFHSTUP name	Description
Temporary storage strings	is the total number of temporary storage strings specified in the TS= system initialization parameter or in the overrides. The number of strings allocated may exceed the number requested.
Peak number of strings in use	is the peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number.
Times string wait occurred	is the total number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the number of I/O requests, consider increasing the number of strings initially allocated.
Peak number of users waiting on string	is the peak number of I/O requests that were queued at any one time because all strings were in use.
I/O errors on TS data set	is the total number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause.

Terminal autoinstall statistics

This is the DFHSTUP listing for terminals that are connected, while the system is running, by means of the autoinstall facility. These statistics are obtained as **interval**, **end-of-day**, or **requested** statistics. CICS also records **unsolicited** autoinstall statistics, which DFHSTUP prints in a separate report; see "Autoinstalled terminals" on page 335.

Terminal autoinstall: Global statistics

These statistics are available online, and are mapped by the DFHA04DS DSECT.

DFHSTUP name	Field name	Description
Autoinstall attempts	A04VADAT	is the number of eligible autoinstall attempts made during the current session of CICS to create terminal entries as users logged on. For an attempt to be considered eligible, CICS and VTAM must not be terminating, autoinstall must be enabled, and the terminal type must be valid for autoinstall (not pipeline, LU6.1, or LU6.2 parallel sessions). Reset characteristic: reset to zero
Rejected attempts	A04VADRJ	is the number of eligible autoinstall attempts that were subsequently rejected during the current session of CICS. Reasons for rejection can be maximum concurrency value exceeded, invalid bind, the user program has rejected the logon, and so on. If this number is unduly high, check the reasons for rejection. Reset characteristic: reset to zero
Deleted attempts	A04VADLO	is the number of deletions of terminal entries as users logged off during the current session. Reset characteristic: reset to zero
Peak concurrent attempts	A04VADPK	is the highest number of attempts made during the current session to create terminal entries as users logged on at the same time. Reset characteristic: reset to current value

DFHSTUP name	Field name	Description
Times the peak was reached	A04VADPX	is the number of times when the highest number of attempts were made during the current session to create terminal entries as users logged on at the same time. Reset characteristic: reset to 1
Times SETLOGON HOLD issued	A04VADSH	is the number of times that the SETLOGON HOLD command was issued during this run of CICS. CICS issues the VTAM SETLOGON HOLD command when the maximum number of concurrent autoinstall requests allowed (the AIQMAX= system initialization parameter) is exceeded. Reset characteristic: reset to zero
Queued logons	A04VADQT	is the number of attempts that were queued for logon due to delete in progress of the TCTTE for the previous session with the same LU. Reset characteristic: reset to zero
Peak of queued logons	A04VADQK	is the highest number of logons that were queued waiting for TCTTE deletion at any one time. If this is unduly high, consider increasing the delete delay interval parameter of the AILDELAY system initialization parameter. Reset characteristic: reset to current value
Times queued peak reached	A04VADQX	is the number of times this peak was reached. Reset characteristic: reset to 1

Terminal autoinstall: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Autoinstall attempts	is the total number of eligible autoinstall attempts made during the entire CICS session to create terminal entries as users logged on. For an attempt to be considered eligible, CICS and VTAM must not be terminating, autoinstall must be enabled, and the terminal type must be valid for autoinstall (not pipeline, LU6.1, or LU6.2 parallel sessions).
Rejected attempts	is the total number of eligible autoinstall attempts that were subsequently rejected during the entire CICS session. Reasons for rejection can be maximum concurrency value exceeded, invalid bind, the user program has rejected the logon, and so on. If this number is unduly high, check the reasons for rejection.
Deleted attempts	is the total number of deletions of terminal entries as users logged off during the entire session.
Peak concurrent attempts	is the highest number of attempts made during the entire CICS session to create terminal entries as users logged on at the same time.
Times the peak was reached	is the number of times that the “peak concurrent attempts” value was reached during the entire CICS session.
Times SETLOGON HOLD issued	is the number of times that the SETLOGON HOLD command was issued during the entire run of CICS. CICS issues the VTAM SETLOGON HOLD command when the maximum number of concurrent autoinstall requests allowed (the AIQMAX= system initialization parameter) is exceeded.
Queued logons	is the total number of attempts that were queued for logon due to delete in progress of the TCTTE for the previous session with the same LU.
Peak of queued logons	is the highest number of logons that were queued waiting for TCTTE deletion at any one time. If this is unduly high, consider increasing the delete delay interval parameter of the AILDELAY system initialization parameter.

DFHSTUP name	Description
Times queued peak reached	is the number of times that the “peak of queued logons” value was reached.
Shipped delete interval	is the currently-specified time delay, in the form **hhmmss**, between invocations of the timeout delete mechanism that removes redundant shipped terminal definitions. The value is set either by the DSHIPINT system initialization parameter, or by a subsequent SET DELETSHIPPED command.
Shipped delete idle time	is the currently-specified minimum time, in the form **hhmmss**, that an inactive shipped terminal definition must remain installed in this region, before it becomes eligible for removal by the CICS timeout delete mechanism. The value is set either by the DSHIPIDL system initialization parameter, or by a subsequent SET DELETSHIPPED command.
Shipped terminals built	is the number of shipped remote terminal definitions built during the recording period.
Shipped terminals installed	is the number of shipped remote terminal definitions currently installed in this region.
Shipped terminals deleted	is the number of shipped remote terminal definitions deleted during the recording period.
Times interval expired	is the number of times the shipped delete interval between invocations of the timeout delete mechanism expired during the recording period.
Remote deletes received	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions received by this region since the start of the recording period.
Remote deletes issued	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions issued by this region since the start of the recording period.
Successful remote deletes	is the number of shipped terminal definitions deleted from this region because of old-style remote delete instructions, since the start of the recording period.
Current idle count	is the number of shipped terminal definitions that are currently idle.
Current idle time	is the overall total time, for all currently-idle shipped terminal definitions, since each was last used.
Current maximum idle time	is the maximum time since a currently-idle shipped terminal definition was last used.
Total idle count	is the overall total number of times, for all shipped terminal definitions, that each has been idle during the recording period. **Note:** This figure excludes the number of those currently idle (Current idle count).
Total idle time	is the overall total time, for all shipped terminal definitions, for which each has been idle during the recording period. **Note:** This figure excludes the time since currently-idle definitions were last used (Current idle time).
Maximum idle time	is the maximum time for which a shipped terminal definition was idle during the recording period. **Note:** This figure does not take into account the times since currently-idle definitions were last used.

Terminal statistics

This is the DFHSTUP listing for terminal statistics.

There are a number of ways in which terminal statistics are important for performance analysis. From them, you can get the number of inputs and outputs, that is, the loading of the system by end users. Line-transmission faults and transaction faults are shown (these both have a negative influence on performance behavior).

Terminal control: Resource statistics

These statistics are gathered for each terminal, including ISC, IRC and MRO sessions. They are available online, and are mapped by the DFHA06DS DSECT. In addition to this, this DSECT should be used to map the terminal totals record.

DFHSTUP name	Field name	Description
Line Id	A06TETI	is the line number for TCAM and BSAM (sequential device support) lines. The line ID is blank for all other access methods.
		Reset characteristic: not reset
Term Id	A06TETI	is the identifier of each terminal as stated in the TERMINAL keyword in CEDA or in the TRMIDNT= operand in the TCT.
		Reset characteristic: not reset
Luname	A06LUNAM	is the terminal LU name
		Reset characteristic: not reset
The remainder of the information should be used for tracking terminal activity.		
Polls	A06LENP	is the number of polls that have been sent to the terminal. This field is for TCAM and BSAM only.
		Reset characteristic: reset to zero
Terminal Type	A06TETT	is the terminal type as defined in the TCT. For information about terminal types and their codes, see the DFHTCTTE DSECT.
		Reset characteristic: not reset
Acc Meth	A06EAMIB	is the terminal access method as defined in the TCT. For information about access methods and their codes, see the DFHTCTTE DSECT.
		Reset characteristic: not reset
Input Messages	A06TENI	See note.
		Reset characteristic: reset to zero
Output Messages	A06TENO	See note.
		Reset characteristic: reset to zero

Note: Input messages (A06TENI) and output messages (A06TENO) are the amount of message activity per terminal. Input and output messages should represent the message traffic between CICS and the terminal. Input traffic should be the result of operator initiated input: that is, initial transaction input or input as a result of a conversational read to the terminal. Output messages should be output written by the application program or messages sent by CICS.

Input and output messages can vary because of differences in the application program being used on different terminals. ATI-initiated transactions would typically not have terminal input but could result in one or many output messages. A batch oriented terminal could initiate a single transaction that did multiple reads to the terminal resulting in multiple input messages. The differences between the remote and local terminal counts may be a result of different applications that run on them. Otherwise, they should be similar.

DFHSTUP name	Field name	Description
Transactions	A06TEOT	is the number of transactions, both nonconversational and pseudoconversational, that were started at this terminal. The transaction count is less than input messages if conversational transactions are being used. Reset characteristic: reset to zero When the operator signs off, the transaction count is not reset. At this time, message DFHSN1200 is issued containing the transaction count for that operator.
Storage Viols.	A06CSVC	is the number of storage violations that have occurred on this terminal. Reset characteristic: reset to zero
Transmission Errors	A06TETE	is the number of errors for this terminal, or the number of disconnects for this session. Reset characteristic: reset to zero
Transaction Errors	A06TEOE	is the number of transactions associated with this particular terminal that could not be started. This could mean that a transaction identifier has not been defined in the CSD data set, or that the operator does not have the proper security to enter the transaction, or that the transaction has been disabled. Reset characteristic: reset to zero When the operator signs off, the transaction error count is not reset. At this time, message DFHSN1200 is issued containing the transaction error count for that operator.
Pipeline messages—Totals	A06TCNT	is the total throwaway count. Reset characteristic: reset to zero
Pipeline messages—Groups	A06SCNT	is the number of consecutive throwaways. Reset characteristic: reset to zero
Pipeline messages—Max Csec	A06MCNT	is the maximum throwaway count. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	A0GPRTY	is the terminal priority Reset characteristic: not reset
TIOA Storage	A06STG	is the TIOA storage allowed at this terminal. Reset characteristic: reset to zero

Terminal control: Summary resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Line Id	is the line number for TCAM and BSAM (sequential device support) lines. The line ID is blank for all other access methods.
Term Id	is the identifier of each terminal as stated in the TERMINAL keyword in CEDA or in the TRMIDNT= operand in the TCT.
Luname	is the terminal LU name
The remainder of the information should be used for tracking terminal activity.	
Polls	is the total number of polls that have been sent to the terminal. This field is for TCAM and BSAM only.
Terminal Type	is the terminal type as defined in the TCT. For information about terminal types and their codes, see the DFHTCTTE DSECT.
Acc Meth	is the terminal access method as defined in the TCT. For information about access methods and their codes, see the DFHTCTTE DSECT.
Input Messages	See note below.
Output Messages	See note below.

Note: Input and output messages are the amount of message activity per terminal. Input and output messages should represent the message traffic between CICS and the terminal. Input traffic should be the result of operator initiated input: that is, initial transaction input or input as a result of a conversational read to the terminal. Output messages should be output written by the application program or messages sent by CICS.

Input and output messages can vary because of differences in the application program being used on different terminals. ATI-initiated transactions would typically not have terminal input but could result in one or many output messages. A batch oriented terminal could initiate a single transaction that did multiple reads to the terminal resulting in multiple input messages. The differences between the remote and local terminal counts may be a result of different applications that run on them. Otherwise, they should be similar.

Transactions	is the total number of transactions, both nonconversational and pseudoconversational, that were started at this terminal. The transaction count is less than input messages if conversational transactions are being used.
	When the operator signs off, the transaction count is not reset. At this time, message DFHSN1200 is issued containing the transaction count for that operator.
Storage Viols.	is the total number of storage violations that have occurred on this terminal.
Transmission Errors	is the total number of errors recorded for this terminal.
Transaction Errors	is the total number of transactions associated with this particular terminal, that could not be started. This could mean that a transaction identifier has not been defined in the CSD data set, or that the operator does not have the proper security to enter the transaction, or that the transaction has been disabled.
	When the operator signs off, the transaction error count is not reset. At this time, message DFHSN1200 is issued containing the transaction error count for that operator.
Pipeline messages—Totals	is the total throwaway count.
Pipeline messages—Groups	is the total number of consecutive throwaways.
Pipeline messages—Max Csec	is the maximum throwaway count.
TIOA Storage	is the average TIOA storage allowed at this terminal.

Transaction statistics

This is the DFHSTUP listing for transaction statistics.

Transaction statistics: Resource statistics

The transaction statistics show how often each transaction is called.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHXMRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*. In addition to this, this DSECT should be used to map the transaction totals record.

DFHSTUP name	Field name	Description
Trans ID	XMRTI	is the transaction identifier associated with the transaction definition. Reset characteristic: not reset
Program Name	XMRPN	is the name of the initial program to which the transaction linked. Reset characteristic: not reset
Tclass Name	XMRTCL	is the name of the transaction class in which the transaction is defined. Reset characteristic: not reset
Prty	XMRPRTY	is the priority of the transaction, from 0–255. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report. Reset characteristic: not reset
Remote Name	XMRRNAM	is the name of the transaction on the remote system. Reset characteristic: not reset
Remote Sysid	XMRRSYS	is the name of the remote system where the transaction resides. Reset characteristic: not reset
Dynamic	XMRDYN	indicates whether the transaction has been defined as DYNAMIC=YES (Y) or DYNAMIC=NO (N). Reset characteristic: not reset
Attach Count	XMRAC	is the number of times that this transaction has been attached, Reset characteristic: reset to zero
Retry Count	XMRRC	is the number of times that this transaction definition has been used to retry a transaction. Reset characteristic: reset to zero
Dynamic Local	XMRDLC	is the number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further information about dynamic transaction routing, see the programming information in the *CICS/ESA Customization Guide*. Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
Dynamic Remote	XMRDRC	is the number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further guidance about dynamic transaction routing, see the programming information in the *CICS/ESA Customization Guide*. Reset characteristic: reset to zero
Remote Starts	XMRRSC	is the number of attempts to start this transaction on a remote system. This may not necessarily be the same as the number of successful starts. Reset characteristic: reset to zero
Storage Violations	XMRSVC	is the number of storage violations for this transaction that have been detected by CICS storage management. This is a serious concern if it occurs in a production system. You should act immediately to identify the cause of the problem because it can lead to data corruption, and therefore should not be allowed to continue in an operational system. Reset characteristic: reset to zero

Transactions: Summary resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Trans ID	is the transaction identifier associated with the transaction definition.
Program Name	is the name of the initial program to which the transaction was linked.
Tclass Name	is the name of the transaction class in which the transaction is defined.
Prty	is the priority of the transaction, from 1–255. **APAR PN91596** MJO 23/1/97 This field is not totalled in the DFHSTUP report.
Remote Name	is the name of the transaction on the remote system.
Remote Sysid	is the name of the remote system where the transaction resides.
Dynamic	indicates whether the transaction has been defined as DYNAMIC=YES (Y) or DYNAMIC=NO (NO).
Attach Count	is the total number of times this transaction has been attached.
Retry Count	is the total number of times that this transaction definition has been used to retry a transaction.
Dynamic Local	is the total number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further guidance information about dynamic transaction routing. For programming information about dynamic transaction routing, see the *CICS/ESA Customization Guide*.
Dynamic Remote	is the total number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further information about dynamic transaction routing, see the *CICS/ESA Customization Guide*.
Remote Starts	is the total number of times this transaction definition has been used to start a transaction remotely.

DFHSTUP name	Description
Storage Violations	is the total number of storage violations for this transaction that have been detected by CICS storage management. This is a serious concern if it occurs in a production system. You should act immediately to identify the cause of the problem because it can lead to data corruption, and therefore should not be allowed to continue in an operational system.

Transaction class (TCLASS) statistics

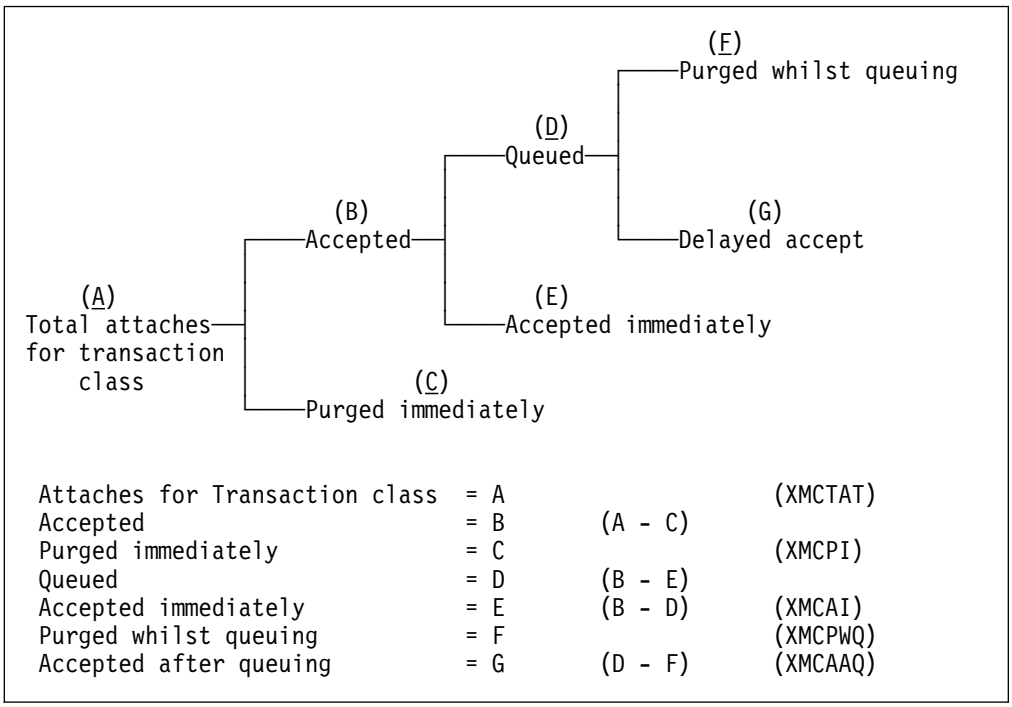
Transaction class: Resource statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHXMCDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

DFHSTUP name	Field name	Description
Tclass Name	XMCTCL	is the 8–character name of the transaction class.
		Reset characteristic: not reset
Number Trandfs	XMCITD	is the number of installed transaction definitions that are defined to belong to this transaction class.
		Note: This will be a reference count from the latest version of the transaction definition table. This statistic is useful to identify redundant tclasses.
		Reset characteristic: not reset
Max Act	XMCMXT	is the maximum number of transactions in the named transaction class that may be active concurrently.
		Reset characteristic: not reset
Purge Thresh	XMCTH	is the queue limit of the purge threshold at which transactions in the named transaction class is purged instead of being added to the queue of transactions that are waiting for membership of the transaction class.
		Reset characteristic: not reset
TOTAL		
–Attaches	XMCTAT	is the total number of attach requests made for transactions in this transaction class.
		Reset characteristic: reset to zero
–AcptImm	XMCAI	is the number of transactions that did not have to queue to become active in this transaction class. They are accepted immediately.
		Reset characteristic: reset to zero
–PrgImm	XMCPI	is the number of transactions that were purged immediately because the queue reached the purge threshold for this transaction class.
		Reset characteristic: reset to zero
–Queued	XMCTQ	is the number of transactions that had to queue for this transaction class.
		Reset characteristic: reset to zero

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	XMCAAQ	is the number of transactions that have become active in this transaction class but queued first.
		Reset characteristic: reset to zero
–PrgQ'd	XMCPWQ	is the number of transactions that have been purged whilst queuing for acceptance into the transaction class. This includes those transactions purged explicitly through Master Terminal, or implicitly through the purge threshold of the transaction class being lowered.
		Reset characteristic: reset to zero
–Q-Time	XMCTQTME	is the total time in STCK units spent waiting by those transactions that were queued in the transaction class.
		Note: This time only includes the time spent by those that have finished queuing. In order to calculate the average queuing time, current queue must be subtracted from the 'queued' count.
		Reset characteristic: reset to zero
Peak Act	XMCPAT	is the highest number of active transactions reached in the transaction class.
		Reset characteristic: reset to current value
Peak Queued	XMCPQT	is the highest number of transactions queued waiting for admittance to the transaction class.
		Reset characteristic: reset to current value
Times MaxAct	XMCTAMA	is the number of separate times that the number of active transactions in the transaction class was equal to the maximum value (XMCMXT). Also registers times when maxactive setting of the tclass is zero and there are no active transactions in the tclass.
		Reset characteristic: reset to zero or one if transaction class is currently at its maxactive limit.
Times PrgThr	XMCTAPT	is the number of separate times that the purge threshold of the transaction class has been reached (times at purge threshold).
		Reset characteristic: reset to zero or one if transaction class is currently at its purge threshold limit.
CURRENT		
–Act	XMCCAT	is the current number of transactions currently active in this transaction class.
		Reset characteristic: not reset
–Queued	XMCCQT	is the number of transactions that are currently queuing in this transaction class.
		Reset characteristic: not reset
–Queue Time	XMCCQTME	is the total time in STCK units spent waiting by those transactions that are currently queuing in this transaction class.
		Reset characteristic: not reset

The following diagram illustrates the transaction class statistics.



Transaction class: Summary resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Tclass Name	is the 8 character name of the transaction class.
Max Act	The maximum number of transactions in the named tclass that may be active concurrently.
Purge Thresh	The queue limit at which transactions in the named tclass will be purged instead of being added to the queue of transactions that are waiting for membership of the transaction class.
Total	
–Attaches	is the total number of attach requests made for transactions in this transaction class.
–AccptImm	The total number of transactions that did not have to queue to become active in this transaction class.
–PurgdImm	The total number of transactions that were purged immediately because they made the queue reach the purge threshold for this transaction class.
–Queued	The total number of transactions that have been made to queue in this transaction class.
–PurgQ'd	The total number of transactions that have been purged whilst queuing for acceptance into the transaction class. This includes those transactions purged explicitly via Master Terminal, or implicitly via the purge threshold of the transaction class being lowered.
–Queuing-Time	The total time spent waiting by those transactions that were queued. Note this time only includes the time spent by those have finished queuing. In order to calculate the average queuing time, current queue must be subtracted from the 'queued' count.
Peak Act	The total highest number of active transactions reached in the transaction class.
Peak Queued	The total highest number of transactions queued waiting for admittance to the transaction class.

DFHSTUP name	Description
Times Max Act	The total number of separate times that the number of active transactions in the transaction class was equal to the maximum value.
Times PurgeThr	The total number of separate times that the purge threshold has been reached.
Average Queuing-Time	The average time spent waiting by those transactions that were queued.

Transaction Manager

Transaction manager: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHXMGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS/ESA System Programming Reference*.

DFHSTUP name	Field name	Description
Total number of transactions (user + system)	XMGNUM	is the number of transactions (user + system) that have run in the system.
		Reset characteristic: reset to zero
Current MAXTASKS limit	XMGMXT	is the latest MXT value (expressed as a number of tasks) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands.
		Reset characteristic: not reset
Current number of active user transactions	XMGCAT	is the current number of active user transactions in the system.
		Reset characteristic: not reset
Current number of queued user transactions	XMGCQT	Current number of queued user transactions in the system. Note that this does not include transactions queueing for tclass membership.
		Reset characteristic: not reset
Times the MAXTASKS limit reached	XMGTAMXT	is the number of times the MXT limit has been reached
		Reset characteristic: reset to zero (or one if at MXT)
Peak number of active user transactions	XMGPAT	is the peak number of active user transactions reached in the system.
		Reset characteristic: reset to current value (XMGCAT)
Number of active user transactions	XMGTAT	is the number of user transactions that have become active.
		Reset characteristic: reset to zero
Number of MAXTASK delayed user transactions	XMGTDT	is the number of user transactions that had to queue for MXT reasons.
		Reset characteristic: reset to zero
Total MAXTASK queuing time	XMGTQTME	is the total time spent waiting by those user transactions that had to queue for MXT reasons.
		Reset characteristic: reset to zero
Total MAXTASK queuing time of currently queued user transactions	XMGCQTME	is the total time spent waiting so far by those user transactions currently queuing for MXT reasons.
		Reset characteristic: not reset

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	XMGTNUM	is the total of user and system transactions attached to date, up to the time of the last statistics reset. **Note:** The total of XMGNUM and XMGTNUM represents the total number of transactions attached so far. Reset characteristic: reset to XMGNUM + XMGTNUM at the time of the last reset.

Transaction manager: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Total number of transactions (user and system)	is the total number of tasks that have run in the system.
MAXTASK limit	is the MXT value (expressed as a number of tasks) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands.
Times the MAXTASK limit reached	is the total number of times MXT has been reached.
Peak number of active user transactions	is the peak number of active user transactions reached in the system.
Total number of active user transactions	is the total number of user transactions that have become active.
Total number of MAXTASK delayed user transactions	is the total number of transactions that had to queue for MXT reasons.
Total MAXTASK queuing time	is the total time spent waiting by those user transactions that had to queue for MXT reasons.
Average MAXTASK queuing time of queued transactions	is the average time spent waiting by those user transactions that had to queue for MXT reasons.

Transient data (global)

Transient data: Global statistics

These statistics are available online, and are mapped by the DFHA11DS DSECT.

DFHSTUP name	Field name	Description
In the statistics produced for the intrapartition data set:		
Control interval size	A11ACISZ	is the size of the control interval, expressed in bytes.
		Reset characteristic: not reset
Control intervals	A11ANCIS	is the current number of control intervals active within the CICS system.
		Reset characteristic: not reset
Peak control intervals used	A11AMXCI	is the peak value of the number of control intervals concurrently in the system.
		Reset characteristic: reset to current value
Times NOSPACE occurred	A11ANOSP	is the number of times that a NOSPACE condition has occurred.
		Reset characteristic: reset to zero
Writes to dataset	A11ACTPT	is the number of WRITEs to the transient data data set. This includes both WRITEs needed for recovery (see below) and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation.
		Reset characteristic: reset to zero
Reads from dataset	A11ACTGT	is the number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
		Reset characteristic: reset to zero
Formatting writes	A11ACTFT	is the number of times a new CI was written at the end of the data set in order to increase the amount of available space.
		Reset characteristic: reset to zero
I/O errors	A11ACTIO	is the number of input/output errors that have occurred during this run of CICS.
		Reset characteristic: reset to zero
In the statistics produced for buffer usage:		
Intrapartition buffers	A11ANBFA	is the number of transient data buffers specified in the system initialization table (SIT) or in the SIT overrides. The number of buffers allocated may exceed the number requested.
		Reset characteristic: not reset
Peak intra. buffers containing valid data	A11AMXIU	is the peak number of intrapartition buffers which contain valid data.
		Reset characteristic: reset to current value
Intrapartition accesses	A11ATNAL	is the number of times intrapartition buffers have been accessed.
		Reset characteristic: reset to current value
Peak concurrent intrapartition accesses	A11AMXAL	is the peak value of the number of concurrent intrapartition buffer accesses.
		Reset characteristic: reset to current value

DFHSTUP name	Field name	Description
Intrapartition buffer waits	A11ATNWT	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.
		Reset characteristic: reset to current value
Peak intrapartition buffer waits	A11AMXWT	is the peak number of requests queued because no buffers were available.
		Reset characteristic: reset to current value
All of the intrapartition data set statistics above are printed, even if the values reported are zero.		
CICS produces the following statistics for multiple strings:		
Number of strings	A11SNSTA	is the number of strings currently active.
		Reset characteristic: not reset
Times string accessed	A11STNAL	is the number of times a string was accessed.
		Reset characteristic: reset to current value
Peak concurrent string accesses	A11SMXAL	is the peak number of strings concurrently accessed in the system.
		Reset characteristic: reset to current value
String waits	A11STNWT	is the number of times that tasks had to wait due to no strings being available.
		Reset characteristic: reset to current value
Peak string waits	A11SMXWT	is the peak number of concurrent string waits in the system.
		Reset characteristic: reset to current value

Transient data: Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
In the statistics produced for the intrapartition data set:	
Control interval size	is the last value encountered for the size of the control interval, expressed in bytes.
Peak control intervals used	is the peak number of control intervals concurrently in the system.
Times NOSPACE occurred	is a total number of times that a NOSPACE condition has occurred.
WRITEs to dataset	is the total number of WRITEs to the temporary storage data set. This includes both WRITEs needed for recovery (see below) and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation.
READs from dataset	is the total number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
Formatting WRITEs	is the total number of times a new CI was written at the end of the data set in order to increase the amount of available space.
I/O errors	is the total number of input/output errors that have occurred during this run of CICS.
In the statistics produced for buffer usage:	
Intrapartition buffers	is the last value encountered for the number of transient data buffers specified by the TD system initialization parameter. The number of buffers allocated may exceed the number requested.
Peak intra. buffers containing valid data	is the peak number of intrapartition data sets which contain valid data.
Intrapartition accesses	is the total number of times that intrapartition data sets have been accessed.
Peak concurrent intrapartition accesses	is the peak number of concurrent intrapartition data set accesses.
Intrapartition buffer waits	is the total number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.
Peak intrapartition buffer waits	is the peak number of requests queued because no buffers were available.
In the statistics produced for the intrapartition data set:	
All of the intrapartition data set statistics above are printed, even if the values reported are zero.	
CICS produces the following statistics for multiple strings:	
Times strings accessed	is the total number of times a string was accessed.
Peak concurrent string accesses	is the peak number of strings concurrently accessed in the system.
String waits	is the total number of times that tasks had to wait due to no strings being available.
Peak string waits	is the peak number of concurrent string waits in the system.

Transient data: Resource statistics

These statistics are collected for each destination. You can use the information from the statistics for each destination to calculate the average number of transient data accesses per transaction. The items in this listing reflect the information you placed in the destination control table (DCT). The statistics are available online, and are mapped by the DFHA10DS DSECT. In addition, this DSECT should be used to map the transient data totals record.

DFHSTUP name	Field name	Description
Dest Id	A10DEST	is the destination identifier (called a “queue” by the application programmer) that you specified in the DESTID operand of the DFHDCT macro. Reset characteristic: not reset
Remote Name	A10RQID	The remote queue name (for a TYPE=REMOTE queue), as specified via the TYPE=REMOTE, RMTNAME operand of the DFHDCT macro. Reset characteristic: not reset.
Remote Sysid	A10RSID	The remote system name (for a TYPE=REMOTE queue), as specified via the TYPE=REMOTE, SYSIDNT operand of the DFHDCT macro. Reset characteristic: not reset.
Indirect Name	A10IDQN	The indirect destination queue name (for a TYPE=INDIRECT queue), as specified via the TYPE=INDIRECT, DESTID operand of the DFHDCT macro. Reset characteristic: not reset.
Only one of the following four fields contains valid data, depending on the class of destination.		
Extra. Outputs	A10EO	is the number of WRITEs to the output data set or READs from the input data set. Reset characteristic: reset to zero
Intra. Outputs	A10IO	is the number of WRITEs to an intrapartition data set. This includes queues defined with a trigger level. Reset characteristic: reset to zero
Indirect Requests	A10IR	is the number of WRITEs to or READs from an indirect destination. Reset characteristic: reset to zero
Remote Requests	A10RR	is the number of READs and WRITEs made to a remote destination identifier. The remote destination can be defined to be across LU6.2, LU6.1, and MRO connections. Reset characteristic: reset to zero
NOT IN DFHSTUP REPORT	A10TYPE	is the destination type. • X'01' Extrapartition queues • X'02' Intrapartition queues • X'03' Indirect queues • X'04' Remote queues Reset characteristic: reset to zero

Note: The above statistics are also totalled for all destinations.

Transient data: Summary resource statistics

Summary statistics are not available online.

DFHSTUP name	Description
Dest Id	is the destination identifier (called a “queue” by the application programmer) that you specified in the DESTID operand of the DFHDCT macro.
Remote Queue Name	The remote queue name (for a TYPE=REMOTE queue), as specified via the TYPE=REMOTE, RMTNAME operand of the DFHDCT macro.
Remote Sysid	The remote system name (for a TYPE=REMOTE queue), as specified via the TYPE=REMOTE, SYSIDNT operand of the DFHDCT macro.
Indirect Name	The indirect destination queue name (for a TYPE=INDIRECT queue), as specified via the TYPE=INDIRECT, DESTID operand of the DFHDCT macro.
Only one of the following five fields contains valid data, depending on the class of destination.	
Extra. Outputs	is the total number of WRITEs to the output data set or READs from the input data set.
Intra. Outputs	is the total number of WRITEs to an intrapartition data set. This includes queues defined with a trigger level.
Indirect Requests	is the total number of WRITEs to or READs from an indirect destination.
Remote Requests	is the total number of READs and WRITES made to a remote destination identifier. The remote destination can be defined to be across LU6.2, LU6.1, and MRO connections.

Note: The above statistics are also totalled for all destinations.

User domain statistics

These statistics are not available online, and are mapped by the DFHUSGDS DSECT.

User domain: global statistics

DFHSTUP name	Field name	Description
Timeout mean reuse time	USGTOMRT	the average time user instances remain on the timeout queue until they are removed.
		Reset characteristic: reset to zero
Timeout reuse count	USGTORC	the number of times a user instance is reused while it was being timed out.
		Reset characteristic: reset to zero
Timeout expiry count	USGTOEC	the number of times a user instance remains on the timeout queue for a full USRDELAY interval without being reused, and is deleted.
		Reset characteristic: reset to zero
Directory reuse count	USGDRRC	the number of times a directory entry was reused.
		Reset characteristic: reset to zero
Directory not found count	USGDRNFC	the number of times a user instance was not found in the directory, but was later successfully added.
		Reset characteristic: reset to zero

User domain Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Average timeout reuse time	is the average time user instances remain on the timeout queue until they are removed.
Timeout reuse count	is the number of times a user instance is reused while being timed out.
Timeout expiry count	is the number of times a user instance remains on the timeout queue for a full USRDELAY interval without being reused.
Directory reuse count	records how many times an existing user instance is reused.
Directory not found count	records the number of times the user instance needs to be added if it doesnot already exist in the directory.

VTAM statistics: Global statistics

These statistics are available online, and are mapped by the DFHA03DS DSECT.

DFHSTUP name	Field name	Description
Times at RPL maximum	A03RPLXT	is the number of times the peak RPL posted value (A03RPLX) was reached. Reset characteristic: reset to zero
Peak RPLs posted	A03RPLX	is the maximum number of receive-any request parameter lists (RPLs) that are posted by VTAM on any one dispatch of terminal control. Reset characteristic: reset to zero
Short on storage count	A03VTSOS	is a counter that is incremented in the VTAM SYNAD exit in the CICS terminal control program each time VTAM indicates that there is a temporary VTAM storage problem. Reset characteristic: reset to zero
Dynamic opens count	A03DOC	is the number of times the VTAM access method control block (ACB) was opened through the control terminal. If VTAM is started before CICS and stays active for the whole CICS run, this value is zero. Reset characteristic: reset to zero
Current LUs in session	A03LUNUM	is the current number of LUs in session. The types of LU that are included are: • LU6.1 primaries and secondaries in session (bound) • LU6.2 primaries and secondaries in session (bound) • VTAM terminals. Reset characteristic: not reset.
HWM LUs in session	A03LUHWM	is the current highest number of LUs logged on. The types of LU that are included are: • LU6.1 primaries and secondaries in session (bound) • LU6.2 primaries and secondaries in session (bound) • VTAM terminals. Reset characteristic: reset to current value.
PS inquire count	A03PSIC	is the number of times CICS issued INQUIRE OPTCD=PERSESS. Reset characteristic: reset to current value.
PS nib count	A03PSNC	is the number of VTAM sessions that persisted. Reset characteristic: reset to current value.
PS opndst count	A03PSOC	is the number of persisting sessions that were successfully restored. Reset characteristic: reset to current value.
PS unbind count	A03PSUC	is the number of persisting sessions that were terminated. Reset characteristic: reset to current value.
PS error count	A03PSEC	is the number of persisting sessions that were already unbound when CICS tried to restore them. Reset characteristic: reset to current value.

VTAM Summary global statistics

Summary statistics are not available online.

DFHSTUP name	Description
Times at RPL maximum	is the total number of times the maximum RPL posted value (A03RPLX) was reached.
Peak RPLs posted	is the peak number of receive-any request parameter lists (RPLs) that are posted by VTAM on any one dispatch of terminal control.
Short on storage count	is a counter that is incremented in the VTAM SYNAD exit in the CICS terminal control program each time VTAM indicates that there is a temporary VTAM storage problem.
Dynamic opens count	is the total number of times that the VTAM access method control block (ACB) was opened through the control terminal. If VTAM is started before CICS and stays active for the whole CICS run, this value is 0.
Current LUs in session	is the average value for the number of LUs logged on.
HWM LUs in session	is the highest value of the number of LUs logged on.
# PS inquire count	is the total number of times CICS issued INQUIRE OPTCD=PERSESS.
# PS nib count	is the total number of VTAM sessions that persisted.
# PS opndst count	is the total number of persisting sessions that were successfully restored.
# PS unbind count	is the total number of persisting sessions that were terminated.
# PS error count #	is the total number of persisting sessions that were already unbound when CICS tried to restore them.

End of Product-Sensitive programming interface

Appendix B. The sample statistics program, DFH0STAT

You can use the statistics sample program, DFH0STAT, to help you determine and adjust the values needed for CICS/ESA storage parameters, for example, using DSALIMIT and EDSALIMIT. The program produces a report showing critical system parameters from the CICS/ESA dispatcher, an analysis of the CICS/ESA storage manager and loader statistics, and an overview of the MVS storage in use. The program demonstrates the use of the EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of a CICS/ESA system. You can use the sample program as provided or modify it to suit your needs.

The sample statistics program consists of the following resources:

DFH0STAT
Statistics sample program, VS COBOL II release 2 or later.

DFH$STAS
Assembler language program called by DFH0STAT.

DFH$STCN
Assembler language program called by DFH0STAT.

DFH0STM
Mapset used by the STAT transaction.

STAT
Transaction used to invoke the program, DFH0STAT.

All programs are command level and run above the 16MB line.

DFH0STAT can be invoked from the PLT at PLTPI (second phase) or PLTSD (first phase) or as a CICS transaction (either from a console or as a conversational transaction from a terminal). The output is sent via the CICS JES SPOOL interface for which a number of default parameters can be changed by the user to specify the distribution of the report(s). These defaults are defined in the working-storage section of this program under the 01 level “OUTPUT-DEFAULTS”.

If an EXEC CICS SPOOL .. command fails when the program is run as a transaction, an error message is displayed on the users screen and the transaction will continue. If the program is not being run from a terminal, a message will be sent to the console using EXEC CICS WRITE OPERATOR commands and the transaction will be terminated normally.

To enable you to use the sample program, you must:

1. Assemble and link-edit the BMS mapset DFH0STM. Include the physical mapset in a library that is in the DFHRPL concatenation. You can either include the symbolic mapset in a user copy library or insert it directly into the source of DFH0STAT.
2. Translate the DFH0STAT program source code, turning CICS commands into code understood by the compiler. The program source code is provided in CICS410.SDFHSAMP.

   Note: You must use the translator option SP when translating DFH0STAT.
3. Compile the translator output for DFH0STAT to produce object code.

4. Link-edit the object module to produce a load module, which you store in an application load library that is concatenated to the DFHRPL DD statement of the CICS/ESA startup job stream.
5. Create resource definition entries, in the CSD, for the programs, mapset, and the STAT transaction.
6. Define the system initialization parameter SPOOL=YES. This specifies that you need support for the system spooling interface.

Analyzing CICS/ESA 4.1 DFH0STAT Reports

DFH0STAT can produce reports about:

- System Status, Monitoring and Statistics
- Transaction Manager and Dispatcher
- Storage
- Loader
- Transactions
- Transaction Totals including Subspace usage information
- Programs
- Program Totals
- DFHRPL Analysis
- Temporary Storage
- Transient Data
- LSR Pools
- Files
- Data Tables.

The heading of each report includes the generic applid, sysid, jobname, date and time, and the CICS/ESA version and release information.

System Status Report

Figure 33 on page 443 shows the format of the System Status Report. The field headings and contents are described in Table 14 on page 444.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37         CICS/ESA Version 4 Release 1.0    PAGE     1


System Status

  MVS Release . . . . . :  MVS/SP4.2.2

  CICS Startup. . . . . :  WARM
  CICS Status . . . . . :  ACTIVE

  Storage Protection. . . :  INACTIVE

  Transaction Isolation . :  INACTIVE

  Reentrant Programs. . . :  PROTECT


Monitoring

  Monitoring . . . . . :  ON
  Exception Class. . . :  ON
  Performance Class. . :  ON
  Sysevent Class . . . :  OFF

  Subsystem id . . . . :  IYAH

  Exception Class Records . . . . . :          0
  Exception Records Suppressed. . . :          0
  Performance Class Records . . . . :         33
  Performance Records Suppressed. . :          0
  SMF Records . . . . . :          3
  SMF Errors. . . . . :          0
  SYSEVENT Records. . . . . :          0
  SYSEVENT Errors . . . . . :          0


Statistics

  Statistics End-of-Day Time . . :  00:00:00
  Statistics Interval. . . . . :  01:00:00
  Next Statistics Collection . . :  09:00:00
  Statistics Recording . . . . . :  ON
```

Figure 33. The System Status Report

Table 14 (Page 1 of 2). Fields in the System Status Report	
Field Heading	**Description**
System Status	
MVS Release	identifies the level of MVS. Source field: CVTPRODN
CICS Startup	indicates the type of CICS startup. Source field: EXEC CICS INQUIRE SYSTEM STARTUP(cvda)
CICS Status	indicates the current status of the local CICS system. Source field: EXEC CICS INQUIRE SYSTEM CICSSTATUS(cvda)
Storage Protection	indicates the status of storage protection. Source field: EXEC CICS INQUIRE SYSTEM STOREPROTECT(cvda)
Transaction Isolation	indicates the status of transaction isolation. Source field: SMSTRANISO
Reentrant Programs	indicates if read-only programs reside in key-0 protected storage. Source field: SMSRENTPGM
Monitoring	
Monitoring	indicates whether CICS monitoring is active in the system. Source field: EXEC CICS INQUIRE MONITOR STATUS(cvda)
Exception Class	indicates whether the exception class of CICS monitoring data is being collected. Source field: EXEC CICS INQUIRE MONITOR EXCEPTCLASS(cvda)
Performance Class	indicates whether the performance class of CICS monitoring data is being collected. Source field: EXEC CICS INQUIRE MONITOR PERFCLASS(cvda)
Sysevent Class	indicates whether the sysevent class of CICS monitoring data is being collected. Source field: EXEC CICS INQUIRE MONITOR EVENTCLASS(cvda)
Subsystem ID	is the 4-character name used as the subsystem identification in the monitoring sysevent class records. Source field: EXEC CICS INQUIRE MONITOR SUBSYSTEMID(4-character data-area)
Exception Class Records	is the number of exception records written to SMF. Source field: MNGER
Exception Class Suppressed	is the number of exception records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGERS
Performance Class Records	is the number of performance records scheduled for output to SMF. Because the monitoring domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered. Source field: MNGPR

Table 14 (Page 1 of 2). Fields in the System Status Report	
Field Heading	**Description**
Performance Class Suppressed	is the number of performance records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGPRS
SMF Records	is the number of SMF records written to the SMF data set. CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing. Source field: MNGSMFR
SMF Errors	is the number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive. Source field: MNGSMFE
SYSEVENT Records	is the number of SYSEVENT notification records written to the MVS SRM (for later processing by RMF). A sysevent record is written at the completion of a transaction. Source field: MNGSYSER
SYSEVENT Errors	is the number of non-OK responses from the request to write a record to the MVS SRM (for later processing by RMF). This count is incremented when a SYSEVENT write fails for any reason, for example, when RMF in inactive. Source field: MNGSYSEE
Statistics	
Statistics End-of-Day Time	is the current end-of-day time for recording statistics. Source field: EXEC CICS INQUIRE STATISTICS ENDOFDAY
Statistics Interval	is the current statistics recording interval. Source field: EXEC CICS INQUIRE STATISTICS INTERVAL
Next Statistics Collection	is the next statistics recording time. Source field: EXEC CICS INQUIRE STATISTICS NEXTTIME
Statistics Recording	is the current status of statistics recording. Source field: EXEC CICS INQUIRE STATISTICS RECORDING(cvda)

Transaction Manager and Dispatcher Report

Figure 34 on page 446 shows the format of the Transaction Manager and Dispatcher Report. The field headings and contents are described in Table 15 on page 447.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37         CICS/ESA Version 4 Release 1.0    PAGE     2

Transaction Manager

  Accumulated transactions so far. . . . . :              49

  Maximum transactions allowed (MXT) . . . :              51
  Times at MXT . . . . . . . . . . . . . . :               0
  Current Active User transactions . . . . :               1
  Peak Active User transactions. . . . . . :               3
  Total Active User transactions . . . . . :              20

  Current Running transactions . . . . . . :               1
  Current Dispatchable transactions. . . . :               0
  Current Suspended transactions . . . . . :               0
  Current System transactions. . . . . . . :               0

  Transactions Delayed by MXT. . . . . . . :               0
  Total MXT queueing time. . . . . . . . . :  00:00:00.00000
  Average MXT queueing time. . . . . . . . :  00:00:00.00000

  Current Queued User transactions . . . . :               0
  Total Queueing time for current queued . :  00:00:00.00000
  Average Queueing time for current queued :  00:00:00.00000

Dispatcher

  Dispatcher start time. . . :  07:36:19.59131

  Peak tasks . . . . . . . . :      23
  Current tasks. . . . . . . :      18


  Current ICV time . . . . . :       5,000ms
  Current ICVR time. . . . . :       5,000ms
  Current ICVTSD time. . . . :         500ms
  Current PRTYAGING time . . :      32,768ms


  Number of active CICS TCBs :       3


   TCB       TCB          TCB         Op. System    Op. System         TCB             TCB             DS TCB
   Name      Status     Start Time       Waits       Wait Time     Dispatch Time     CPU Time        CPU Time
  QR_SUBD    Active   07:36:19.59131         797  00:01:09.81784  00:00:07.48087  00:00:01.94567  00:00:00.12920
  RO_SUBD    Active   07:36:26.73323         126  00:00:33.71868  00:00:36.43828  00:00:02.64071  00:00:00.12556
  CO_SUBD    Inactive
  SZ         Active   07:36:19.59178          60  00:01:17.18938  00:00:00.10917  00:00:00.09120  00:00:00.01155
  Totals                                                          00:00:44.02833  00:00:04.67760  00:00:00.26632
```

Figure 34. The Transaction Manager and Dispatcher Report

Table 15 (Page 1 of 3). Fields in the Transaction Manager and Dispatcher Report	
Field Heading	**Description**
Transaction Manager	
Accumulated transactions so far	is the number of tasks that have accumulated. Source field: XMGNUM
Maximum transactions allowed (MXT)	is the specified maximum number of user transactions as specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands. Source field: XMGMXT
Times at MXT	is the number of times that the number of active user transactions equalled the specified maximum number of user transactions (MXT). Source field: XMGTAMXT
Current Active User transactions	is the current number of active user transactions. Source field: XMGCAT
Peak Active User transactions	is the peak number of active user transactions reached. Source field: XMGPAT
Total Active User transactions	is the total number of user transactions that have become active. Source field: XMGTAT
Current Running transactions	is the current number of Running transactions. Source field: EXEC CICS INQUIRE TASKLIST RUNNING
Current Dispatchable transactions	is the current number of Dispatchable transactions. Source field: EXEC CICS INQUIRE TASKLIST DISPATCHABLE
Current Suspended transactions	is the current number of Suspended transactions. Source field: EXEC CICS INQUIRE TASKLIST SUSPENDED
Current System transactions	is the current number of system transactions. Source field: ((Running + Dispatchable + Suspended) - XMGCAT)
Transactions Delayed by MXT	is the number of user transactions that had to queue for MXT reasons before becoming active, excluding those still waiting. Source field: XMGTDT
Total MXT Queueing Time	is the total time spent waiting by those user transactions that had to wait for MXT reasons. **Note:** This does not include those transactions still waiting. Source field: XMGTQTME
Average MXT Queueing Time	is the average time spent waiting by those user transactions that had to wait for MXT reasons. Source field: (XMGTQTME / XMGTDT)
Current Queued User transactions	is the current number of user transactions currently queuing for MXT reasons. **Note:** That this does not include transactions currently queued for Transaction Class. Source field: XMGCQT

Table 15 (Page 1 of 3). Fields in the Transaction Manager and Dispatcher Report	
Field Heading	**Description**
Total Queueing Time for current queued	is the total time spent waiting by those user transactions currently queued for MXT reasons. **Note:** This does not include the time spent waiting by those transactions that have finished queuing. Source field: XMGCQTME
Average Queueing Time for current queued	is the average time spent waiting by those user transactions currently queued for MXT reasons. Source field: (XMGCQTME / XMGCQT)
Dispatcher	
Dispatcher Start Time	is the time at which the dispatcher started. Source field: DSGSTART
Peak Tasks	is the peak number of tasks concurrently in the system. Source field: DSGPNT
Current tasks	is the current number of tasks in the system. This figure includes all system tasks and all user tasks. Source field: DSGCNT
Current ICV Time	is the ICV time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands. Source field: DSGICVT
Current ICVR Time	is the current task runaway time interval. Source field: EXEC CICS INQUIRE SYSTEM RUNAWAY
Current ICVTSD Time	is the ICVTSD time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands. Source field: DSGICVSD
Current PRTYAGING Time	is the current task priority aging factor. Source field: EXEC CICS INQUIRE SYSTEM PRTYAGING
Number of active CICS TCBs	is the number of active CICS TCBs. Source field: DSGTCBF1 -> DSGTCBAC
TCB Name	is the name of the TCB that the statistics refer to. The names are:- • 'QR_SUBD' • 'RO_SUBD' • 'CO_SUBD' • 'SZ ' Source field: DSGTCBNM
TCB Status	is the current status of this CICS dispatcher TCB. **Note:** If the TCB is inactive there are no statistics in the following fields. Source field: DSGTCBAC

Table 15 (Page 1 of 3). Fields in the Transaction Manager and Dispatcher Report	
Field Heading	**Description**
TCB Start Time	is the time at which the dispatcher started this TCB. Source field: DSGSTART
Op. System Waits	is the number of MVS waits which occurred on this TCB. Source field: DSGSYSW
Op. System Wait Time	is the accumulated real time that this TCB was in an MVS wait, that is, the total time used between an MVS wait issued by the dispatcher and the return from the MVS wait. Source field: DSGTWT
TCB Dispatch Time	is the accumulated real time that this TCB has been dispatched by MVS, that is, the total time used between an MVS wait issued by the dispatcher and the subsequent wait issued by the dispatcher. Source field: DSGTDT
TCB CPU Time	is the accumulated CPU time taken for this TCB, that is, the total time that this TCB has been in execution. Source field: DSGACT
DS TCB CPU Time	is the accumulated CPU time taken for this DS task, that is, the processor time used by this TCB while executing the default dispatcher task (DSTCB). Source field: DSGTCT
Totals	
TCB Dispatch Time	is the total accumulated real time that the active TCBs have been dispatched. Source field: DSGTDT for each active TCB
TCB CPU Time	is the total accumulated CPU time taken for the active TCBs. Source field: DSGACT for each active TCB
DS TCB CPU Time	is the total accumulated CPU time taken for the DS task on each active dispatcher TCB. Source field: DSGTCT for each active TCB

Storage Reports

The Storage Below 16Mb Report provides information on the use of MVS and CICS/ESA virtual storage. It contains the information you need to understand your current use of virtual storage below 16Mb and helps you to verify the size values used for the CDSA, UDSA, SDSA and RDSA and the value set for the DSA limit. Figure 35 on page 450 shows the format of the Storage Below 16Mb Report. The field headings and contents are described in Table 16 on page 451.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE     3

  Region size established from REGION= parameter. . . :      73,728K

Storage BELOW 16MB

  Private Area Region size below 16Mb . . . . . :       8,172K
    Max LSQA/SWA storage allocated below 16Mb (SYS) . :       392K
    Max User storage allocated below 16Mb (VIRT). . . :     4,440K
    System Use. . . . . . . . . . . . . . . . . . . . :        20K
    RTM . . . . . . . . . . . . . . . . . . . . . . . . :      250K
  Private Area storage available below 16Mb . . . . . :     3,070K
                                                                    Current DSA Limit . . . . . :     4,096K
                                                                    Current Allocation for DSAs . :     1,792K
  VIRT minus Current DSA Limit. . . . . . . . . . . . :       344K  Peak Allocation for DSAs. . . :     1,792K


                                CDSA          UDSA          SDSA          RDSA          Totals
  Current DSA Size. . . . . . . :          768K          256K          256K          512K          1,792K
  Current DSA Used. . . . . . . :          700K            0K            4K          324K          1,028K
  Current DSA Used as % of DSA. :           91%           0%            1%           63%             57% of DSA Size
* Peak DSA Used . . . . . . . . :          720K            8K            4K          324K
  Peak DSA Size . . . . . . . . :          768K          256K          256K          512K
  Cushion Size. . . . . . . . . :           64K           64K           64K           64K
* Free Storage (inc. Cushion) . :           68K          256K          252K          188K
* Peak Free Storage . . . . . . :          280K          256K          256K          356K
  Lowest Free Storage . . . . . :           48K          248K          252K          188K
  Largest Free Area . . . . . . :           64K          256K          252K          124K
  Largest Free Area as % of DSA :            8%          100%           98%           24%
  Largest Free/Free Storage . . :         0.94          1.00          1.00          0.65

  Current number of extents . . :            3             1             1             2               7
  Number of extents added . . . :            3             1             1             2
  Number of extents released. . :            0             0             0             0

  Getmain Requests. . . . . . . :          268            35             1            21
  Freemain Requests . . . . . . :          116            35             0             0

  Current number of Subpools. . :           33            16             6             3              58
  Add Subpool Requests. . . . . :           67            50             6             3
  Delete Subpool Requests . . . :           34            34             0             0

  Times no storage returned . . :            0             0             0             0
  Times request suspended . . . :            0             0             0             0
  Current requests suspended. . :            0             0             0             0
  Peak requests suspended . . . :            0             0             0             0
  Requests purged while waiting :            0             0             0             0

  Times Cushion released. . . . :            0             0             0             0
  Times Short-On-Storage. . . . :            0             0             0             0
  Total time Short-On-Storage . : 00:00:00.00000  00:00:00.00000  00:00:00.00000  00:00:00.00000
  Average Short-On-Storage time : 00:00:00.00000  00:00:00.00000  00:00:00.00000  00:00:00.00000

  Storage Violations. . . . . . :            0             0             0             0               0

  Access. . . . . . . . . . . . :         CICS          CICS          CICS       READONLY

  '*' indicates values reset on last DSA Size change
```

Figure 35. The Storage Report Below 16Mb

Table 16 (Page 1 of 4). Fields in the Storage Below 16Mb Report	
Field Heading	**Description**
Region size established from REGION= parameter	is the region size established from the REGION= parameter in the JCL. If the region requested was greater than 16 megabytes, the region established resides above 16 megabytes, and this field will be a minimum value of 32 megabytes.
Storage BELOW 16MB	
Private Area Region size below 16Mb	is the private area size below 16MB expressed in Kbytes.
Max LSQA/SWA storage allocated below 16Mb (SYS)	is the maximum amount of virtual storage allocated from the local system queue area (LSQA) and the SWA subpools below 16MB expressed in Kbytes.
Max User storage allocated below 16Mb (VIRT)	is the maximum amount of virtual storage allocated from the user subpools below 16MB expressed in Kbytes.
System Use	is an amount of virtual storage available for system use.
RTM	is an amount of virtual storage available for use by the MVS recovery and termination manager included for calculation purposes, which could be allocated during a CICS region recovery and termination.
Private Area Storage available below 16Mb	is the amount of additional storage that could be allocated by increasing the REGION parameter.
Current DSA Limit	is the current DSA Limit, expressed in Kbytes. Source field: (SMSDSALIMIT / 1024)
Current Allocation for DSAs	is the current amount of storage allocated to the DSAs below 16MB, expressed in Kbytes. This value may be smaller or larger than the current DSA limit. Source field: (SMSDSATOTAL / 1024)
VIRT minus Current DSA Limit	is the total amount of user storage allocated/used below 16MB minus the current DSA limit. Source field: ((VIRT - SMSDSALIMIT) / 1024)
Peak Allocation for DSAs	is the peak amount of storage allocated to the DSAs below 16MB, expressed in Kbytes. This value may be smaller or larger than the current DSA limit. Source field: (SMSHWMDSATOTAL / 1024)
Current DSA Size	is the current size of the CDSA, UDSA, SDSA, or the RDSA, expressed in Kbytes. Source field: (SMSDSASZ / 1024)
Current DSA Used	is the current amount of storage used in this DSA expressed in Kbytes. Source field: ((SMSDSASZ - SMSFSTG) / 1024)
Current DSA Used as % of DSA	is the current amount of storage used in this DSA expressed as a percentage of the current DSA size. Source field: (((SMSDSASZ - SMSFSTG) / SMSDSASZ) * 100)

Table 16 (Page 1 of 4). Fields in the Storage Below 16Mb Report	
Field Heading	**Description**
Peak DSA Used	is the peak amount of storage used in this DSA expressed in Kbytes. Source field: (SMSHWMPS / 1024)
Peak DSA Size	is the peak size of the CDSA, UDSA, SDSA, or the RDSA, expressed in Kbytes. Source field: (SMSHWMDSASZ / 1024)
Cushion Size	is the size of the cushion, expressed in bytes. The cushion forms part of the CDSA, UDSA, SDSA, or the RDSA, and is the amount of storage below which CICS goes SOS. Source field: (SMSCSIZE / 1024)
Free Storage (inc. Cushion)	is the current amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSFSTG / 1024)
Peak Free Storage	is the peak amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSHWMFSTG / 1024)
Lowest Free Storage	is the lowest amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSLWMFSTG / 1024)
Largest Free Area	is the length of the largest contiguous free area in the CDSA, UDSA, SDSA, or RDSA, expressed in bytes. Source field: (SMSLFA / 1024)
Largest Frea Area as % of DSA	is the largest contiguous free area in the CDSA, UDSA, SDSA, or RDSA, expressed as a precentage of the current DSA Size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Frea/Free Storage	is an indication of the storage fragmentation in this pagepool. This value is calculated by dividing the “Largest Free Area” (SMSLFA) by the “Free storage” (SMSFSTG). If the ratio is large, then this pagepool is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	is the current number of extents allocated to this DSA. Source field: SMSEXTS
Number of extents added	is the number of extents added to this DSA. Source field: SMSEXTSA
Number of extents released	is the number of extents released from this DSA. Source field: SMSEXTSR
Getmain Requests	is the number of GETMAIN requests from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSGMREQ

Table 16 (Page 1 of 4). Fields in the Storage Below 16Mb Report	
Field Heading	**Description**
Freemain Requests	is the number of FREEMAIN requests from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSFMREQ
Current number of Subpools	is the current number of subpools (domain and task) in the CDSA, UDSA, SDSA, or RDSA. Source field: SMSCSUBP
Add Subpool Requests	is the number of ADD_SUBPOOL requests to create a subpool (domain or task) from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSASR
Delete Subpool Requests	is the number of DELETE_SUBPOOL requests (domain or task) from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSDSR
Times no storage returned	is the number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS
Times request suspended	is the number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. Source field: SMSUCSS
Current requests suspended	is the number of GETMAIN requests currently suspended for storage. Source field: SMSCSS
Peak requests suspended	is the peak number of GETMAIN requests suspended for storage. Source field: SMSHWMSS
Requests purged while waiting	is the number of requests which were purged while suspended for storage. Source field: SMSPWWS
Times cushion released	is the number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion. Source field: SMSCREL
Times Short-On-Storage	is the number of times CICS went SOS in this pagepool (CDSA, UDSA, SDSA, or RDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. Source field: SMSSOS

Table 16 (Page 2 of 4). Fields in the Storage Below 16Mb Report	
Field Heading	**Description**
Total Short-On-Storage time	is the accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	is the average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)
Storage Violations	is the number of storage violations recorded in the CDSA, UDSA, SDSA, or the RDSA. Source field: SMSSV
Access	is the type of access of the page pool. It will either be CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the RDSA. • CICS - access is CICS key • USER - access is USER key • READONLY - access is read-only protection Source field: SMSACCESS

The Storage Above 16Mb Report provides information on the use of MVS and CICS/ESA virtual storage. It contains the information you need to understand your current use of virtual storage above 16Mb and helps you to verify the size values used for the ECDSA, EUDSA, ESDSA and ERDSA and the value set for the EDSA limit. Figure 36 on page 455 shows the format of the Storage Above 16Mb Report. The field headings and contents are described in Table 17 on page 456.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE     4

Storage ABOVE 16MB

  Private Area Region size above 16Mb . . . . . :   2,015,232K
    Max LSQA/SWA storage allocated above 16Mb (SYS) . :     9,632K
    Max User storage allocated above 16Mb (EXT) . . . :    73,728K
  Private Area storage available above 16Mb . . . . . :   1,931,872K

                                                        Current EDSA Limit. . . . . :      51,200K
  CICS Trace table size . . . . . :      2,000K         Current Allocation for EDSAs. :     9,216K
  EXT minus Current EDSA Limit. . . . . :     22,528K   Peak Allocation for EDSAs . . :     9,216K


                                  ECDSA          EUDSA          ESDSA          ERDSA          Totals
  Current DSA Size. . . . . :       3,072K         1,024K         1,024K         4,096K         9,216K
  Current DSA Used. . . . . :       2,316K            64K             8K         3,324K         5,712K
  Current DSA Used as % of DSA. :      75%             6%             0%            81%            61% of EDSA Size
* Peak DSA Used . . . . . :       2,316K           128K             8K         3,324K
  Peak DSA Size . . . . . :       3,072K         1,024K         1,024K         4,096K
  Cushion Size. . . . . :           128K             0K           128K           256K
  Free Storage (inc. Cushion) . :   756K           960K         1,016K           772K
* Peak Free Storage . . . . . :   1,044K         1,024K         1,024K         1,032K
* Lowest Free Storage . . . . . :   756K           896K         1,016K           772K
  Largest Free Area . . . . . :     756K           960K         1,016K           764K
  Largest Free Area as % of DSA :    24%            93%            99%            18%
  Largest Free/Free Storage . . :  1.00           1.00           1.00           0.98

  Current number of extents . . :       3              1              1              4              9
  Number of extents added . . . :       3              1              1              4
  Number of extents released. . :       0              0              0              0

  Getmain Requests. . . . . :       5,998             96              5            228
  Freemain Requests . . . . . :     1,340             90              0              4

  Current number of Subpools. . :     154             16              3              3            176
  Add Subpool Requests. . . . . :     188             50              3              3
  Delete Subpool Requests . . . :      34             34              0              0

  Times no storage returned . . :       0              0              0              0
  Times request suspended . . . :       0              0              0              0
  Current requests suspended. . :       0              0              0              0
  Peak requests suspended . . . :       0              0              0              0
  Requests purged while waiting :       0              0              0              0

  Times Cushion released. . . . :       0              0              0              0
  Times Short-On-Storage. . . . :       0              0              0              0
  Total time Short-On-Storage . : 00:00:00.00000 00:00:00.00000 00:00:00.00000 00:00:00.00000
  Average Short-On-Storage time : 00:00:00.00000 00:00:00.00000 00:00:00.00000 00:00:00.00000

  Storage Violations. . . . . :         0              0              0              0              0

  Access. . . . . :           CICS           CICS           CICS        READONLY

  '*' indicates values reset on last DSA Size change
```

Figure 36. The Storage Report Above 16Mb

Table 17 (Page 1 of 4). Fields in the Storage Above 16Mb Report	
Field Heading	**Description**
Storage ABOVE 16MB	
Private Area Region size above 16Mb	is the private area size above 16MB expressed in Kbytes.
Max LSQA/SWA storage allocated above 16Mb (SYS)	is the maximum amount of virtual storage allocated from the local system queue area (LSQA) and the SWA subpools above 16MB expressed in Kbytes.
Max User storage allocated above 16Mb (EXT)	is the maximum amount of virtual storage allocated from the user subpools above 16MB expressed in Kbytes.
Private Area Storage available above 16Mb	is the amount of additional storage that could be allocated by increasing the REGION parameter.
Current EDSA Limit	is the current EDSA Limit, expressed in Kbytes. Source field: (SMSEDSALIMIT / 1024)
CICS Trace table size	is the current size of the CICS trace table. Source field: EXEC CICS INQUIRE TRACEDEST TABLESIZE
Current Allocation for EDSAs	is the peak amount of storage allocated to the DSAs above 16MB, expressed in Kbytes. This value may be smaller or larger than the current EDSA limit. Source field: (SMSEDSATOTAL / 1024)
EXT minus Current EDSA Limit	is the total amount of user storage allocated/used above 16MB minus the current DSA limit. Source field: ((EXT - SMSEDSALIMIT) / 1024)
Peak Allocation for EDSAs	is the peak amount of storage allocated to the DSAs above 16MB, expressed in Kbytes. This value may be smaller or larger than the current EDSA limit. Source field: (SMSHWMEDSATOTAL / 1024)
Current DSA Size	is the current size of the ECDSA, EUDSA, ESDSA, or the ERDSA, expressed in Kbytes. Source field: (SMSDSASZ / 1024)
Current DSA Used	is the current amount of storage used in this DSA expressed in Kbytes. Source field: ((SMSDSASZ - SMSFSTG) / 1024)
Current DSA Used as % of DSA	is the current amount of storage used in this DSA expressed as a percentage of the current DSA size. Source field: (((SMSDSASZ - SMSFSTG) / SMSDSASZ) * 100)
Peak DSA Used	is the peak amount of storage used in this DSA expressed in Kbytes. Source field: (SMSHWMPS / 1024)
Peak DSA Size	is the peak size of the ECDSA, EUDSA, ESDSA, or the ERDSA, expressed in Kbytes. Source field: (SMSHWMDSASZ / 1024)

Table 17 (Page 1 of 4). Fields in the Storage Above 16Mb Report	
Field Heading	**Description**
Cushion Size	is the size of the cushion, expressed in bytes. The cushion forms part of the ECDSA, EUDSA, ESDSA, or the ERDSA, and is the amount of storage below which CICS goes SOS. Source field: (SMSCSIZE / 1024)
Free Storage (inc. Cushion)	is the current amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSFSTG / 1024)
Peak Free Storage	is the peak amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSHWMFSTG / 1024)
Lowest Free Storage	is the lowest amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSLWMFSTG / 1024)
Largest Free Area	is the length of the largest contiguous free area in the ECDSA, EUDSA, ESDSA, or ERDSA, expressed in Kbytes. Source field: SMSLFA
Largest Frea Area as % of DSA	is the largest contiguous free area in the ECDSA, EUDSA, ESDSA, or ERDSA, expressed as a precentage of the current DSA Size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Frea/Free Storage	is an indication of the storage fragmentation in this pagepool. This value is calculated by dividing the “Largest Free Area” (SMSLFA) by the “Free storage” (SMSFSTG). If the ratio is large, then this pagepool is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	is the current number of extents allocated to this DSA. Source field: SMSEXTS
Number of extents added	is the number of extents added to this DSA. Source field: SMSEXTSA
Number of extents released	is the number of extents released from this DSA. Source field: SMSEXTSR
Getmain Requests	is the number of GETMAIN requests from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSGMREQ
Freemain Requests	is the number of FREEMAIN requests from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSFMREQ
Current number of Subpools	is the current number of subpools (domain and task) in the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSCSUBP

Table 17 (Page 1 of 4). Fields in the Storage Above 16Mb Report	
Field Heading	**Description**
Add Subpool Requests	is the number of ADD_SUBPOOL requests to create a subpool (domain or task) from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSASR
Delete Subpool Requests	is the number of DELETE_SUBPOOL requests (domain or task) from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSDSR
Times no storage returned	is the number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS
Times request suspended	is the number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. Source field: SMSUCSS
Current requests suspended	is the number of GETMAIN requests currently suspended for storage. Source field: SMSCSS
Peak requests suspended	is the peak number of GETMAIN requests suspended for storage. Source field: SMSHWMSS
Requests purged while waiting	is the number of requests which were purged while suspended for storage. Source field: SMSPWWS
Times cushion released	is the number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion. Source field: SMSCREL
Times Short-On-Storage	is the number of times CICS went SOS in this pagepool (ECDSA, EUDSA, ESDSA, or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. Source field: SMSSOS
Total Short-On-Storage time	is the accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	is the average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)

Table 17 (Page 2 of 4). Fields in the Storage Above 16Mb Report	
Field Heading	**Description**
Storage Violations	is the number of storage violations recorded in the ECDSA, EUDSA, ESDSA, or the ERDSA. Source field: SMSSV
Access	is the type of access of the page pool. It will either be CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the ERDSA. • CICS - access is CICS key • USER - access is USER key • READONLY - access is read-only protection Source field: SMSACCESS

Loader and Program Storage Report

Figure 37 on page 460 shows the format of the Loader and Program Storage Report. The field headings and contents are described in Table 18 on page 461.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE     5

Loader

  Library Load requests. . . . . . . . . . . . . :            295  Total Program Uses . . . . . . . . . . . . . . :            524
  Total Library Load time. . . . . . . . . . . . :  00:00:20.94713  Program Use to Load Ratio. . . . . . . . . . . :           1.77
  Average Library Load time. . . . . . . . . . . :  00:00:00.07099
                                                                    Times DFHRPL secondary extents detected. . . . :              0
  Library Load requests that waited. . . . . . . :              2
  Total Library Load request wait time . . . . . :  00:00:01.10574
  Average Library Load request wait time . . . . :  00:00:00.55286
  Current Waiting Library Load requests. . . . . :              0
  Peak Waiting Library Load requests . . . . . . :              1
  Times at Peak. . . . . . . . . . . . . . . . . :              2  Average Not-In-Use program size. . . . . . . . :             4K


  CDSA                                                              ECDSA

  Programs Removed by compression. . . . . . . . :              0  Programs Removed by compression. . . . . . . . :              0
  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000
  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000
  Programs Reclaimed from the Not-In-Use Queue . :              0  Programs Reclaimed from the Not-In-Use Queue . :             38
  Programs Loaded - on the Not-In-Use Queue. . . :              2  Programs Loaded - now on the Not-In-Use Queue. :              8

  SDSA                                                              ESDSA

  Programs Removed by compression. . . . . . . . :              0  Programs Removed by compression. . . . . . . . :              0
  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000
  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000
  Programs Reclaimed from the Not-In-Use Queue . :              0  Programs Reclaimed from the Not-In-Use Queue . :              4
  Programs Loaded - on the Not-In-Use Queue. . . :              0  Programs Loaded - now on the Not-In-Use Queue. :              5

  RDSA                                                              ERDSA

  Programs Removed by compression. . . . . . . . :              0  Programs Removed by compression. . . . . . . . :              0
  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000
  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000
  Programs Reclaimed from the Not-In-Use Queue . :              0  Programs Reclaimed from the Not-In-Use Queue . :              4
  Programs Loaded - on the Not-In-Use Queue. . . :              2  Programs Loaded - now on the Not-In-Use Queue. :             15


Program Storage

  Nucleus Program Storage (CDSA) . . . . . . . . :            80K  Nucleus Program Storage (ECDSA). . . . . . . . :             64K

  Program Storage (SDSA) . . . . . . . . . . . . :             0K  Program Storage (ESDSA). . . . . . . . . . . . :              8K
  Resident Program Storage (SDSA). . . . . . . . :             0K  Resident Program Storage (ESDSA) . . . . . . . :              0K

  Read-Only Nucleus Program Storage (RDSA) . . . :           156K  Read-Only Nucleus Program Storage (ERDSA). . . :          2,776K
  Read-Only Program Storage (RDSA) . . . . . . . :           168K  Read-Only Program Storage (ERDSA). . . . . . . :            548K
  Read-Only Resident Program Storage (RDSA). . . :             0K  Read-Only Resident Program Storage (ERDSA) . . :              0K

  CDSA used by Not-In-Use programs. :         21K   2.70% of CDSA   ECDSA used by Not-In-Use programs :         15K   0.49% of ECDSA
  SDSA used by Not-In-Use programs. :          0K   0.00% of SDSA   ESDSA used by Not-In-Use programs :          8K   0.78% of ESDSA
  RDSA used by Not-In-Use programs. :          2K   0.30% of RDSA   ERDSA used by Not-In-Use programs :        103K   2.52% of ERDSA
```

Figure 37. The Loader and Program Storage Report

Table 18 (Page 1 of 4). Fields in the Loader and Program Storage Report	
Field Heading	**Description**
Loader	
Library Load requests	is the number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure. Source field: LDGLLR
Total Program Uses	is the number of uses of any program by the CICS system. Source field: LDGPUSES
Total Library Load time	is the time taken for the number of library loads indicated by LDGLLR. Source field: LDGLLT
Program Use to Load Ratio	is the ratio of program uses to programs loads. Source field: (LDGPUSES / LDGLLR)
Average Library Load time	is the average time to load a program. Source field: (LDGLLT / LDGLLR)
Times DFHRPL secondary extents detected	is the number of times the loader received an end-of-extent condition during a LOAD and successfully closed and reopened the DFHRPL library and retried the LOAD. Source field: LDGDREBS
Library Load requests that waited	is the number of loader domain requests that ***were*** forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress. This figure is the total number of tasks that have waited, and does not include those that are currently waiting (LDGWLR). Source field: LDGWTDLR
Total Library Load request wait time	is the suspended time for the number of tasks indicated by LDGWTDLR. Source field: LDGTTW
Average Library Load request wait time	is the average loader domain request suspend time. Source field: (LDGTTW / LDGWTDLR)
Current Waiting Library Load requests	is the number of loader domain requests that ***are currently*** forced to suspend due to the loader domain currently performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress. Source field: LDGWLR

Table 18 (Page 1 of 4). Fields in the Loader and Program Storage Report	
Field Heading	**Description**
Peak Waiting Library Load requests	is the maximum number of tasks suspended at one time. Source field: LDGWLRHW
Times at Peak	is the number of times the high watermark level indicated by LDGWLRHW was reached. This, along with the previous two values, is an indication of the level of contention for loader resource. Source field: LDGHWMT
Average Not-In-Use program size	is the average size of a program currently on the Not-In-Use queue. Source field: ((LDGCNIU + LDGSNIU + LDGRNIUS + LDGECNIU + LDGESNIU + LDGERNIU) / 1024) / LDGPROGNIU)
Programs Removed by compression	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Source field: LDGDPSCR
Time on the Not-In-Use Queue	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. Source field: LDGDPSCT
Average Time on the Not-In-Use Queue	is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. Source field: (LDGDPSCT / LDGDPSCR)
Programs Reclaimed from the Not-In-Use Queue	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Source field: LDGRECNIU
Programs Loaded - on the Not-In-Use Queue	is the number of programs on the Not-In-Use (NIU) queue. Source field: LDGPROGNIU
Program Storage	

Table 18 (Page 1 of 4). Fields in the Loader and Program Storage Report

Field Heading	Description
Nucleus Program Storage (CDSA)	is the current amount of storage allocated to nucleus programs in the CDSA. **APAR PQ07674** MJO 26/08/98 Source field: (SMDCPS for subpool 'LDNUC ' and 'LDNRS ' / 1024)
Nucleus Program Storage (ECDSA)	is the current amount of storage allocated to nucleus programs in the ECDSA. Source field: (SMDCPS for subpool 'LDENUC ' and 'LDENRS ' / 1024)
Program Storage (SDSA)	is the current amount of storage allocated to programs in the SDSA. Source field: (SMDCPS for subpool 'LDPGM ' / 1024)
Program Storage (ESDSA)	is the current amount of storage allocated to programs in the ESDSA. Source field: (SMDCPS for subpool 'LDEPGM ' / 1024)
Resident Program Storage (SDSA)	is the current amount of storage allocated to resident programs in the SDSA. Source field: (SMDCPS for subpool 'LDRES ' / 1024)
Resident Program Storage (ESDSA)	is the current amount of storage allocated to resident programs in the ESDSA. Source field: (SMDCPS for subpool 'LDERES ' / 1024)
Read-Only Nucleus Program Storage (RDSA)	is the current amount of storage allocated to nucleus programs in the RDSA. Source field: (SMDCPS for subpool 'LDNUCRO ' and 'LDNUCRO ' / 1024)
Read-Only Nucleus Program Storage (ERDSA)	is the current amount of storage allocated to nucleus programs in the ERDSA. Source field: (SMDCPS for subpool 'LDENUCRO' and 'LDENRSRO' / 1024)
Read-Only Program Storage (RDSA)	is the current amount of storage allocated to programs in the RDSA. Source field: (SMDCPS for subpool 'LDPGMRO ' / 1024)
Read-Only Program Storage (ERDSA)	is the current amount of storage allocated to programs in the ERDSA. Source field: (SMDCPS for subpool 'LDEPGMRO' / 1024)

Table 18 (Page 2 of 4). Fields in the Loader and Program Storage Report	
Field Heading	**Description**
Read-Only Resident Program Storage (RDSA)	is the current amount of storage allocated to resident programs in the RDSA. Source field: (SMDCPS for subpool 'LDRESRO ' / 1024)
Read-Only Resident Program Storage (ERDSA)	is the current amount of storage allocated to resident programs in the ERDSA. Source field: (SMDCPS for subpool 'LDERESRO' / 1024)
CDSA used by Not-In-Use programs	is the current amount of CDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(1) / 1024)
ECDSA used by Not-In-Use programs	is the current amount of ECDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(2) / 1024)
SDSA used by Not-In-Use programs	is the current amount of UDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(3) / 1024)
ESDSA used by Not-In-Use programs	is the current amount of EUDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(4) / 1024)
RDSA used by Not-In-Use programs	is the current amount of UDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(5) / 1024)
ERDSA used by Not-In-Use programs	is the current amount of ERDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(6) / 1024)

Transactions Report

Figure 38 on page 465 shows the format of the Transactions Report. The field headings and contents are described in Table 19 on page 466.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE     6

Transactions

Tran    Tran    Program                          Task Data      Attach      Restart     Dynamic  -  Counts        Remote
 id     Class     Name     Dynamic  Isolate  Location/Key     Count        Count       Lo cal      Remote        Starts

AADD            DFH$AALL  Static     Yes    Below/USER           0           0           0           0             0
ABRW            DFH$ABRW  Static     Yes    Below/USER           0           0           0           0             0
ADMC            ADMPSTBC  Static     Yes    Below/USER           0           0           0           0             0
ADMI            ADMISSEC  Static     Yes    Below/USER           0           0           0           0             0
ADMM            ADM1IMDC  Static     Yes    Below/USER           0           0           0           0             0
ADMP            ADMOPUC   Static     Yes    Below/USER           0           0           0           0             0
ADMU            ADM5IVUC  Static     Yes    Below/USER           0           0           0           0             0
ADMV            ADMVSSEC  Static     Yes    Below/USER           0           0           0           0             0
ADM4            ADM4CDUC  Static     Yes    Below/USER           0           0           0           0             0
ADYN            DFH99     Static     Yes    Below/USER           0           0           0           0             0
AINQ            DFH$AALL  Static     Yes    Below/USER           0           0           0           0             0
AMNU            DFH$AMNU  Static     Yes    Below/USER           0           0           0           0             0
AORD            DFH$AREN  Static     Yes    Below/USER           0           0           0           0             0
AORQ            DFH$ACOM  Static     Yes    Below/USER           0           0           0           0             0
APPA            APPCP05   Dynamic    Yes      Any/USER           0           0           0           0             0
APPC            APPCP00   Dynamic    Yes      Any/USER           0           0           0           0             0
AREP            DFH$AREP  Static     Yes    Below/USER           0           0           0           0             0
AUPD            DFH$AALL  Static     Yes    Below/USER           0           0           0           0             0
BLOG            BLOGPROG  Dynamic    Yes    Below/USER           0           0           0           0             0
CALL            CALL      Dynamic    Yes    Below/CICS           0           0           0           0             0
CATA            DFHZATA   Static     Yes      Any/CICS           1           0           0           0             0
CATD            DFHZATD   Static     Yes      Any/CICS           0           0           0           0             0
CATR            DFHZATR   Static     Yes      Any/CICS           0           0           0           0             0
CBRC            DFHBRCP   Static     Yes    Below/CICS           0           0           0           0             0
CDBC            DFHDBME   Static     Yes      Any/CICS           0           0           0           0             0
CDBD            DFHDBDI   Static     Yes      Any/CICS           0           0           0           0             0
CDBI            DFHDBIQ   Static     Yes      Any/CICS           0           0           0           0             0
CDBN            DFHDBCON  Static     Yes      Any/CICS           0           0           0           0             0
CDBO            DFHDBCT   Static     Yes      Any/CICS           0           0           0           0             0
CDBT            DFHDBDSC  Static     Yes      Any/CICS           0           0           0           0             0
CDTS            DFHZATS   Static     Yes      Any/CICS           0           0           0           0             0
CEBR            DFHEDFBR  Static     Yes      Any/CICS           0           0           0           0             0
CECI            DFHECIP   Static     Yes    Below/USER           1           0           0           0             0
CECS            DFHECSP   Static     Yes      Any/CICS           0           0           0           0             0
CEDA            DFHEDAP   Static     Yes      Any/CICS           0           0           0           0             0
CEDB            DFHEDAP   Static     Yes      Any/CICS           0           0           0           0             0
CEDC            DFHEDAP   Static     Yes      Any/CICS           0           0           0           0             0
CEDF            DFHEDFP   Static     Yes      Any/CICS           0           0           0           0             0
CEHP            DFHCHS    Static     Yes    Below/CICS           0           0           0           0             0
CEHS            DFHCHS    Static     Yes    Below/CICS           0           0           0           0             0
CEMT            DFHEMTP   Static     Yes    Below/CICS           0           0           0           0             0
CEOT            DFHEOTP   Static     Yes    Below/CICS           0           0           0           0             0
CESC            DFHCESC   Static      No      Any/CICS           0           0           0           0             0
CESF            DFHSFP    Static     Yes    Below/CICS           0           0           0           0             0
CESN            DFHSNP    Static     Yes    Below/CICS           0           0           0           0             0
CEST            DFHESTP   Static     Yes    Below/CICS           0           0           0           0             0
CETR            DFHCETRA  Static     Yes    Below/CICS           0           0           0           0             0
CFTS            DFHZATS   Static     Yes      Any/CICS           0           0           0           0             0
CGRP            DFHZCGRP  Static      No      Any/CICS           0           0           0           0             0
CITS            DFHZATS   Static     Yes      Any/CICS           0           0           0           0             0
```

Figure 38. The Transactions Report

Table 19. Fields in the Transactions Report	
Field Heading	**Description**
Transactions	
Tran id	is the name of the transaction. Source field: EXEC CICS INQUIRE TRANSACTION
Tran Class	is the name of the transaction class in which the transaction is defined. Source field: XMRTCL
Program Name	is the name of the program when the transaction was defined, or spaces if a program name was not supplied. Source field: XMMRPN
Dynamic	indicates whether the transaction was defined as dynamic. Source field: XMRDYN
Isolate	indicates whether the transaction's user-key task-lifetime storage is isolated from the user-key programs of other transactions. Source field: EXEC CICS INQUIRE TRANSACTION ISOLATEST
Task Data Location	is where certain CICS control blocks will be located for the transaction. Source field: EXEC CICS INQUIRE TRANSACTION TASKDATALOC
Task Data Key	is the storage key in which CICS will obtain all storage for use by the transaction. Source field: EXEC CICS INQUIRE TRANSACTION TASKDATAKEY
Attach Count	is the number of times this transaction definition has been used to attach a transaction. Source field: XMRAC
Restart Count	is the number of times this transaction was restarted after an abend. This field is zero if the transaction was not defined as RESTART=YES. Source field: XMRRC
Dynamic Counts - Local	is the total number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. Source field: XMRDLC
Dynamic Counts - Remote	is the total number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. Source field: XMRDRC
Remote Starts	is the total number of times this transaction definition has been used to start a transaction remotely. Source field: XMRRSC

Transaction Totals Report

Figure 39 on page 467 shows the format of the Transactions Totals Report. The field headings and contents are described in Table 20 on page 467.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE     9

Transaction Totals

            Task Data    Subspace  Transaction    Attach
 Isolate  Location/Key   Usage       Count        Count

    Yes     Below/CICS     None          31            0
    Yes       Any/CICS     None          51           15
    Yes     Below/USER    Unique         41            1
    Yes       Any/USER    Unique         27            0
     No     Below/CICS    Common          3            1
     No       Any/CICS    Common          4            0
     No     Below/USER    Common          0            0
     No       Any/USER    Common          1            1
Totals                                   158           18

Subspace Statistics

 Current Unique Subspace Users (Isolate=Yes). . . :          0
 Peak Unique Subspace Users (Isolate=Yes) . . . . :          0
 Total Unique Subspace Users (Isolate=Yes). . . . :          0

 Current Common Subspace Users (Isolate=No) . . . :          0
 Peak Common Subspace Users (Isolate=No). . . . . :          0
 Total Common Subspace Users (Isolate=No) . . . . :          0
```

Figure 39. The Transaction Totals Report

Table 20 (Page 1 of 2). Fields in the Transaction Totals Report

Field Heading	Description
Transaction Totals	
Isolate	indicates whether the transaction's user-key task-lifetime storage is isolated from the user-key programs of other transactions.
Task Data Location/Key	is the number of transactions defined with this combination of task data location and task data key.
Subspace Usage	indicates the type of subspace usage for these transaction definitions.
Transaction Count	is the number of transaction definitions for this combination of isolate, task data location, task data key, and subspace usage.
Attach Count	is the number of times these transaction definitions have been used to attach a transaction.
Subspace Statistics	
Current Unique Subspace Users (Isolate=Yes)	is the current number of tasks allocated a unique subspace. Source field: SMSUSSCUR
Peak Unique Subspace Users (Isolate=Yes)	is the peak number of tasks allocated a unique subspace. Source field: SMSUSSHWM

Table 20 (Page 1 of 2). Fields in the Transaction Totals Report	
Field Heading	**Description**
Total Unique Subspace Users (Isolate=Yes)	is the total number of tasks that have been allocated a unique subspace. Source field: SMSUSSCUM
Current Common Subspace Users (Isolate=No)	is the current number of tasks allocated to the common subspace. Source field: SMSCSSCUR
Peak Common Subspace Users (Isolate=No)	is the peak number of tasks allocated to the common subspace. Source field: SMSCSSHWM
Total Common Subspace Users (Isolate=No)	is the total number of tasks that have been allocated to the common subspace. Source field: SMSCSSCUM

Programs Report

Figure 40 on page 469 shows the format of the Programs Report. The field headings and contents are described in Table 21 on page 470.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE    10

Programs

Program   Data   Exec                Times          Total          Average       RPL      Times      Times     Program     Program
 Name      Loc   Key   Times Used    Fetched      Fetch Time     F etch Time     Offset   Newcopy    Removed     Size      Location

DFH$STAT   Any   CICS           1          1   00:00:00.07291   00:00:00.07291      0          0          0     98,832    ERDSA
DFH$STCN   Any   CICS          49          1   00:00:00.00534   00:00:00.00534      0          0          0      1,232    ECDSA
DFH$STEX   Any   CICS           0          0                                       1          0          0        136    None
DFH$STVS   Any   CICS           0          0                                       1          0          0      6,032    None
DFH$TDWT   Below USER           0          0                                       4          0          0        488    None
DFH$ZCAT   Any   USER           0          0                                       1          0          0         32    None
DFHACP     Any   CICS           0          0                                       4          0          0      8,216    ECDSA
DFHAKP     Below CICS           0          0                                       4          0          0      5,672    CDSA
DFHAMP     Any   CICS           0          0                                       4          0          0    108,712    None
DFHAPATT   Any   CICS           1          0                                       4          0          0        632    ERDSA
DFHBRCP    Below CICS           0          0                                       4          0          0     14,824    None
DFHCCNV    Any   CICS           0          0                                       4          0          0      6,088    None
DFHCESC    Any   CICS           0                                                  0          0                        None
DFHCETRA   Any   CICS           0          0                                       4          0          0     11,928    None
DFHCETRB   Any   CICS           0          0                                       4          0          0     13,680    None
DFHCETRC   Any   CICS           0          0                                       4          0          0     12,560    None
DFHCETRD   Any   CICS           0          0                                       4          0          0      4,088    None
DFHCHS     Below CICS           0          0                                       4          0          0      8,168    None
DFHCLS4    Any   CICS           0          0                                       4          0          0      4,624    None
DFHCMAC    Any   CICS           0          0                                       4          0          0      7,656    None
DFHCMCM                         0          0                                       4          0          0      2,840    None
DFHCNV     Below CICS           0                                                  0          0                        None
DFHCPY     Any   CICS           0          0                                       4          0          0        320    None
DFHCRNP    Any   CICS           0          0                                       4          0          0     10,008    ERDSA
DFHCRQ     Any   CICS           0          0                                       4          0          0        856    ERDSA
DFHCRR     Any   CICS           0          0                                       4          0          0      1,928    ERDSA
DFHCRS     Any   CICS           0          0                                       4          0          0      5,344    None
DFHCRSP    Any   CICS           0          0                                       4          0          0      3,176    ERDSA
DFHCRT     Any   CICS           0          0                                       4          0          0        640    None
DFHCSSC    Any   CICS           0          0                                       4          0          0      3,984    None
DFHCTRH                         0          0                                       4          0          0     13,512    None
DFHCTRM                         0          0                                       4          0          0      4,344    None
DFHCWTO    Any   CICS           0          0                                       4          0          0      1,784    None
DFHCXCU    Any   CICS           0          0                                       4          0          0        680    None
DFHDBAT    Below CICS           0          0                                       4          0          0      7,744    None
DFHDBCON   Any   CICS           0          0                                       4          0          0      6,568    None
DFHDBCT    Any   CICS           0          0                                       4          0          0     16,328    None
DFHDBDI    Any   CICS           0          0                                       4          0          0      1,736    None
DFHDBDSC   Any   CICS           0          0                                       4          0          0      2,320    None
DFHDBIE                         0          0                                       4          0          0      1,368    None
DFHDBIQ    Any   CICS           0          0                                       4          0          0      3,624    None
DFHDBME    Any   CICS           0          0                                       4          0          0     10,608    None
DFHDBMS    Any   CICS           0          0                                       4          0          0        880    None
DFHDBNE                         0          0                                       4          0          0      2,744    None
DFHDBP1$   Any   CICS           0          0                                       4          0          0      4,608    ERDSA
DFHDBP2$   Below CICS           0          0                                       4          0          0      5,344    None
DFHDBTI    Below CICS           0          0                                       4          0          0      8,440    None
DFHDBUEX   Any   CICS           0          0                                       4          0          0        272    None
DFHDLG     Below CICS           0          0                                       4          0          0      1,504    None
DFHDLRP    Below CICS           0          0                                       4          0          0        960    RDSA
DFHDLS     Below CICS           0          0                                       4          0          0        600    None
DFHDMP     Any   CICS           0          0                                       4          0          0     41,944    None
```

Figure 40. The Programs Report

Table 21 (Page 1 of 2). Fields in the Programs Report	
Field Heading	**Description**
Programs	
Program Name	is the name of the program. Source field: EXEC CICS INQUIRE PROGRAM
Data Loc	is the storage location which the program is able to accept. Source field: EXEC CICS INQUIRE PROGRAM DATALOCATION
EXEC Key	is the access key in which the program will execute. Source field: EXEC CICS INQUIRE PROGRAM EXECKEY
Times Used	is the number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue an MVS LOAD. Source field: LDRTU
Times Fetched	is the number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage. Source field: LDRFC
Total Fetch Time	is the time taken to perform all fetches for this program. Source field: LDRFT
Average Fetch Time	is the average time taken to perform a fetch of the program. Source field: (LDRFT / LDRFC)
RPL Offset	is the offset into the DFHRPL DD concatenation of the library from which the program was last loaded or is loaded when next required non-LPA resident modules only. **Note:** The offset values begin with zero for the first partitioned data set in the concatenation and thus this field may not be used to deduce whether a copy of the program is available to the loader domain. Source field: LDRRPLO
Times Newcopy	is the number of times a NEWCOPY has been requested against this program. Source field: LDRTN
Times Removed	is the number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism. Source field: LDRRPC
Program Size	is the size of the program in bytes, if known (otherwise zero). Source field: LDRPSIZE

Table 21 (Page 1 of 2). Fields in the Programs Report	
Field Heading	**Description**
Program Location	is the location of the current storage resident instance of the program, if any. It has one of the following values: • None - No current copy • CDSA - Current copy is in the CDSA • SDSA - Current copy is in the SDSA • RDSA - Current copy is in the RDSA • ECDSA - Current copy is in the ECDSA • ESDSA - Current copy is in the ESDSA • ERDSA - Current copy is in the ERDSA • LPA - Current copy is in the LPA • ELPA - Current copy is in the ELPA Source field: LDRLOCN

Program Totals Report

Figure 41 on page 472 shows the format of the Program Totals Report. The field headings and contents are described in Table 22 on page 473.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37         CICS/ESA Version 4 Release 1.0    PAGE    24

Program Totals

 Programs. . . . . :         611
   Assembler . . . . :         564
   C . . . . . :             2
   COBOL . . . . . :          43
   LE/370. . . . . :           0
   PL1 . . . . . :             2
   Other . . . . . :           0
 Maps. . . . . :            93
 Partitionsets . . . . :       1
 Total . . . . . :         705


 CDSA Programs . . . . :       4
 SDSA Programs . . . . :       0
 RDSA Programs . . . . :       7

 ECDSA Programs. . . . :      22
 ESDSA Programs. . . . :       5
 ERDSA Programs. . . . :      36

 LPA Programs. . . . . :       0
 ELPA Programs . . . . :       0

 Unused Programs . . . :      284
 Not Located Programs. :      347
 Total . . . . . :         705
```

Figure 41. The Program Totals Report

Table 22. Fields in the Program Totals Report	
Field Heading	**Description**
Program Totals	
Programs - Assembler	is the current total number of programs defined to CICS as Assembler programs.
Programs - C	is the current total number of programs defined to CICS as C programs.
Programs - COBOL	is the current total number of programs defined to CICS as COBOL programs.
Programs - LE/370	is the current total number of programs defined to CICS as LE/370 programs.
Programs - PL1	is the current total number of programs defined to CICS as PL/1 programs.
Programs - Other	is the current total number of programs defined to CICS that are not Assembler, C, COBOL, LE/370 or PL/1.
Maps	is the current number of maps defined to CICS.
Partitionsets	is the current number of partitionsets defined to CICS.
Total	is the total number of programs, maps, and partitionsets defined to CICS.
CDSA Programs	is the current number of programs, maps, and partitionsets defined to CICS residing in the CDSA.
SDSA Programs	is the current number of programs, maps, and partitionsets defined to CICS residing in the SDSA.
RDSA Programs	is the current number of programs, maps, and partitionsets defined to CICS residing in the RDSA.
ECDSA Programs	is the current number of programs, maps, and partitionsets defined to CICS residing in the ECDSA.
ESDSA Programs	is the current number of programs, maps, and partitionsets defined to CICS residing in the ESDSA.
ERDSA Programs	is the current number of programs, maps, and partitionsets defined to CICS residing in the ERDSA.
LPA Programs	is the current number of programs, maps, and partitionsets defined to CICS residing in the LPA.
ELPA Programs	is the current number of programs, maps, and partitionsets defined to CICS residing in the ELPA.
Unused Programs	is the current number of programs, maps, and partitionsets defined to CICS and which have been located in a concatenation of the DFHRPL DD program library but which have not been used by any CICS task.
Not Found Programs	is the current number of programs, maps, and partitionsets defined to CICS but which have not been located in any concatenation of the DFHRPL DD program library.
Total	is the total number of programs, maps, and partitionsets defined to CICS.

DFHRPL Analysis Report

Figure 42 on page 474 shows the format of the DFHRPL Analysis Report. The field headings and contents are described in Table 23 on page 474.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE    25

DFHRPL Analysis

  RPL                                          Average
 Offset   Programs   Times Used    Fetches     Fetch Time     Newcopies    Removes

    0          27           51          3  00:00:00.03046           0           0
    1          86            2          1  00:00:00.03070           0           0
    2          79           52          0                           0           0
    3           1            0          0                           0           0
    4         161           14          8  00:00:00.01857           0           0
    5           1            0          0                           0           0
    6           1            0          0                           0           0
    7           1            0          0                           0           0
    8           1            0          0                           0           0
Totals        358          119         12                           0           0
```

Figure 42. The DFHRPL Analysis Report

Table 23. Fields in the DFHRPL Analysis Report

Field Heading	Description
DFHRPL Analysis	
RPL Offset	is the offset into the DFHRPL DD program library concatenation.
Programs	is the current number of programs, maps, and partitionsets defined to CICS and located in this concatenation of the DFHRPL DD program library.
Times Used	is the number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program that have fetched from this concatenation of the DFHRPL DD program library. Source field: LDRTU
Fetches	is the number of times programs were fetched from this concatenation of the DFHRPL DD program library. Source field: LDRFC
Average Fetch Time	is the average fetch time for programs fetched from this concatenation of the DFHRPL DD program library. Source field: (LDRFT / LDRFC)
Newcopies	is the number of times programs were newcopied which have been fetched from this concatenation of the DFHRPL DD program library. Source field: LDRTN
Removes	is the number of times programs were removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism which had been fetched from this concatenation of the DFHRPL DD program library. Source field: LDRRPC

Temporary Storage Report

Figure 43 on page 475 shows the format of the Temporary Storage Report. The field headings and contents are described in Table 24 on page 475.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE    27

Temporary Storage

  Put/Putq main storage requests . . . . . :           6
  Get/Getq main storage requests . . . . . :          18
  Peak storage used for TS Main. . . . . :           0K
  Current storage used for TS Main . . . . :           0K

  Put/Putq auxiliary storage requests. . . :         318
  Get/Getq auxiliary storage requests. . . :          12

  Times temporary storage queue created. . :           0
  Peak temporary storage queues in use . . :           6
  Current temporary storage queues in use. :           6
  Items in longest queue . . . . . :       1,150
  Queue extension threshold. . . . . :          20
  Queue extensions created . . . . . :          16

  Control interval size. . . . . :       4,096
  Control intervals in the DFHTEMP dataset :         300
  Peak control intervals used. . . . . :          39
  Current control intervals in use . . . . :          39
  Available bytes per control interval . . :       4,032
  Segments per control interval. . . . . :          63
  Bytes per segment. . . . . :          64
  Writes bigger than control interval size :           0
  Largest record length written. . . . . :         186
  Times auxiliary storage exhausted. . . . :           0
  Number Temporary storage compressions. . :           8

  Temporary storage strings. . . . . :           4
  Peak Temporary storage strings in use. . :           1
  Temporary storage string waits . . . . . :           0
  Peak users waiting on string . . . . . :           0
  Current users waiting on string. . . . . :           0

  Temporary storage buffers. . . . . :          18
  Temporary storage buffer waits . . . . . :           0
  Peak users waiting on buffer . . . . . :           0
  Current users waiting on buffer. . . . . :           0
  Temporary storage buffer reads . . . . . :           0
  Temporary storage buffer writes. . . . . :          16
  Forced buffer writes for recovery. . . . :           7
  Format writes. . . . . :           0

  I/O errors on the DFHTEMP dataset. . . . :           0
```

Figure 43. The Temporary Storage Report

Table 24 (Page 1 of 4). Fields in the Temporary Storage Report

Field Heading	Description
Temporary Storage	
Put/Putq main storage requests	is the number of records that application programs wrote to main temporary storage. Source field: A12STA5F
Get/Getq main storage requests	is the number of records that application programs obtained from main temporary storage. Source field: A12NMG

Table 24 (Page 1 of 4). Fields in the Temporary Storage Report	
Field Heading	**Description**
Peak storage used for TS Main	is the peak value, expressed in Kbytes, of the amount of virtual storage used for temporary storage records. Source field: (A12STA6F / 1024)
Current storage used for TS Main	is the current value, expressed in Kbytes, of the amount of virtual storage used for temporary storage records. Source field: (A12STA6A / 1024)
Put/Putq auxiliary storage requests	is the number of records that application programs wrote to auxiliary temporary storage. Source field: A12STA7F
Get/Getq auxiliary storage requests	is the number of records that application programs obtained from auxiliary temporary storage. Source field: A12NAG
Times temporary storage queue created	is the number of times that CICS created individual temporary storage queues. Source field: A12STA3F
Peak temporary storage queues in use	is the peak number of temporary storage queue names in use at any one time. Source field: A12QNUMH
Current temporary storage queues in use	is the current number of temporary storage queue names in use. Source field: A12QNUM
Items in longest queue	is the peak number of items in any one temporary storage queue. Source field: A12QINH
Queue extension threshold	is the number of records that are held in a single temporary storage group identifier (TSGID). This value is controlled by the system initialization parameter, TSMSGSET. Source field: A12GIDNE
Queue extensions created	is the number of times it was necessary to create a TSGID extension. The capacity of a single TSGID is determined by the TSMGSET option in the SIT, and indirectly controls how many additional extension blocks have to be built. Source field: A12STA4F
Control interval size	is the size of VSAM's unit of transmission between DASD and main storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set. In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead. Source field: A12CSZ
Control intervals in the DFHTEMP dataset	is the number of control intervals (CIs) available for auxiliary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination. Source field: A12NCI

Table 24 (Page 1 of 4). Fields in the Temporary Storage Report	
Field Heading	**Description**
Peak control intervals in use	is the peak number of control intervals (CIs) containing active data. Source field: A12NCIAH
Current control intervals in use	is the current number of control intervals (CIs) containing active data. Source field: A12NCIA
Available bytes per control interval	is the number of bytes available for use in a DFHTEMP data set control interval (CI). Source field: A12NAVB
Segments per control interval	is the number of segments available in a DFHTEMP data set control interval (CI). Source field: A12SPCI
Bytes per segment	is the number of bytes per segment of the DFHTEMP data set. Source field: A12BPSEG
Writes bigger than control interval size	is the number of writes of records whose length was greater than the control interval (CI) size. Source field: A12STABF
Largest record length written	is the size, expressed in bytes, of the longest record written to the temporary storage data set. Source field: A12LAR
Times auxiliary storage exhausted	is the number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced to abend. Source field: A12STA8F
Number Temporary Storage compressions	is the number of times that the temporary storage buffers were compressed. Source field: A12STA9F
Temporary storage strings	is the number of temporary storage strings specified in the TS= system initialization parameter or in the overrides. The number of strings allocated may exceed the number requested. Source field: A12NVCA
Peak Temporary storage strings in use	is the peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number. Source field: A12NVCAH
Temporary storage string waits	is the number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the number of I/O requests, consider increasing the number of strings initially allocated. Source field: A12VWTN

Table 24 (Page 2 of 4). Fields in the Temporary Storage Report	
Field Heading	**Description**
Peak users waiting on string	is the peak number of I/O requests that were queued at any one time because all strings were in use. Source field: A12VUWTH
Current users waiting on string	is the current number of I/O requests that are queued because all strings are in use. Source field: A12VUWT
Temporary storage buffers	is the number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides. The number of buffers allocated may exceed the number requested. Source field: A12NBCA
Temporary storage buffer waits	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Source field: A12BWTN
Peak users waiting on buffer	is the peak number of requests queued because no buffers were available. Source field: A12BUWTH
Current users waiting on buffer	is the current number of requests queued because no buffers are available. Source field: A12BUWT
Temporary storage buffer reads	is the number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. Source field: A12TRDN
Temporary storage buffer writes	is the number of WRITEs to the temporary storage data set. This includes both WRITEs necessitated by recovery requirements (see next item) and WRITEs forced by the buffer being needed to accommodate another CI. Source field: A12TWTN
Forced buffer writes for recovery	is the subset of the total number of WRITEs caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation. Source field: A12TWTNR
Format writes	is the number of times a new control interval (CI) was successfully written at the end of the data set to increase the amount of available space in the temporary storage data set. A formatted write is attempted only if the current number of control intervals (CIs) available in the auxiliary data set have all been used. Source field: A12TWTNF
I/O errors on the DFHTEMP dataset	is the number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause. Source field: A12STAAF

Transient Data Report

Figure 44 on page 479 shows the format of the Transient Data Report. The field headings and contents are described in Table 25 on page 479.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE    27

Transient Data

  Transient data reads . . . . . :          0
  Transient data writes. . . . . :          0
  Transient data formatting writes . . . . :          0

  Control interval size. . . . . :      1,536
  Control intervals in the DFHINTRA dataset:        115
  Peak control intervals used. . . . . :          4
  Times NOSPACE on DFHINTRA occurred . . . :          0

  Transient data strings . . . . . :          3
  Times Transient data string in use . . . :          0
  Peak Transient data strings in use . . . :          0
  Times string wait occurred . . . . . :          0
  Peak users waiting on string . . . . . :          0

  Transient data buffers . . . . . :          5
  Times Transient data buffer in use . . . :        324
  Peak Transient data buffers in use . . . :          1
  Peak buffers containing valid data . . . :          3
  Times buffer wait occurred . . . . . :          0
  Peak users waiting on buffer . . . . . :          0

  I/O errors on the DFHINTRA dataset . . . :          0
```

Figure 44. The Transient Data Report

Table 25 (Page 1 of 2). Fields in the Transient Data Report

Field Heading	Description
Transient Data	
Transient data reads	is the number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. Source field: A11ACTGT
Transient data writes	is the number of WRITEs to the transient data data set. This includes both WRITEs needed for recovery (see below) and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation. Source field: A11ACTPT
Transient data formatting writes	is the number of times a new CI was written at the end of the data set in order to increase the amount of available space. Source field: A11ACTFT
Control interval size	is the size of the control interval, expressed in bytes. Source field: A11ACISZ
Control intervals in the DFHINTRA dataset	is the current number of control intervals active within the CICS system. Source field: A11ANCIS
Peak control intervals used	is the peak value of the number of control intervals concurrently in the system. Source field: A11AMXCI

Table 25 (Page 1 of 2). Fields in the Transient Data Report	
Field Heading	**Description**
Times NOSPACE on DFHINTRA occurred	is the number of times that a NOSPACE condition has occurred. Source field: A11ANOSP
Transient data strings	is the number of strings currently active. Source field: A11STSTA
Times Transient data string in use	is the number of times a string was accessed. Source field: A11STNAL
Peak Transient data strings in use	is the peak number of strings concurrently accessed in the system. Source field: A11SMXAL
Times string wait occurred	is the number of times that tasks had to wait due to no strings being available. Source field: A11STNWT
Peak users waiting on string	is the peak number of concurrent string waits in the system. Source field: A11SMXWT
Transient data buffers	is the number of transient data buffers specified in the system initialization table (SIT) or in the SIT overrides. The number of buffers allocated may exceed the number requested. Source field: A11ANBFA
Times Transient data buffer in use	is the number of times intrapartition buffers have been accessed. Source field: A11ATNAL
Peak Transient data buffers in use	is the peak value of the number of concurrent intrapartition buffer accesses. Source field: A11AMXAL
Peak buffers containing valid data	is the peak number of intrapartition buffers which contain valid data. Source field: A11AMXIU
Times buffer wait occurred	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Source field: A11ATNWT
Peak users waiting on buffer	is the peak number of requests queued because no buffers were available. Source field: A11AMXWT
I/O errors on the DFHINTRA dataset	is the number of input/output errors that have occurred on the DFHINTRA dataset during this run of CICS. Source field: A11ACTIO

LSR Pools Report

Figure 45 on page 481 shows the format of the LSR Pools Report. The field headings and contents are described in Table 26 on page 482.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE    22

LSR Pools

  Pool Number :   2     Time Created :   10:10:21.65472

     Maximum key length . . . . . :          16
     Total number of strings  . . . . :       5
     Peak concurrently active strings :       1
     Total requests waited for string :       0
     Peak requests waited for string. :       0

  Buffer Totals

     Data Buffers . . . . . :          44          Index Buffers. . . . . :          44
     Hiperspace Data Buffers. . . . . :   0          Hiperspace Index Buffers . . . . :   0
       Successful look asides . . . . :  652          Successful look asides . . . . :  1,360
       Buffer reads . . . . . :          24          Buffer reads . . . . . :          4
       User initiated writes. . . . . :  655          User initiated writes. . . . . :  31
       Non-user initiated writes. . . :  0            Non-user initiated writes. . . :  0
       Successful Hiperspace CREADS . :  0            Successful Hiperspace CREADS . :  0
       Successful Hiperspace CWRITES. :  0            Successful Hiperspace CWRITES. :  0
       Failing Hiperspace CREADS. . . :  0            Failing Hiperspace CREADS. . . :  0
       Failing Hiperspace CWRITES . . :  0            Failing Hiperspace CWRITES . . :  0

  Data Buffer Statistics

                    Hiperspace    Look                    User     Non-User  Successful  Successful   Failing     Failing
    Size  Buffers     Buffers     Asides      Reads      Writes     Writes     CREADS      CWRITES     CREADS      CWRITES

     512      4           0          0          0          0          0          0           0           0           0
    1024      4           0          0          0          0          0          0           0           0           0
    2048      4           0          0          0          0          0          0           0           0           0
    4096      4           0        652         24        655          0          0           0           0           0
    8192      4           0          0          0          0          0          0           0           0           0
   12288      4           0          0          0          0          0          0           0           0           0
   16384      4           0          0          0          0          0          0           0           0           0
   20480      4           0          0          0          0          0          0           0           0           0
   24576      4           0          0          0          0          0          0           0           0           0
   28672      4           0          0          0          0          0          0           0           0           0
   32768      4           0          0          0          0          0          0           0           0           0

  Index Buffer Statistics

                    Hiperspace    Look                    User     Non-User  Successful  Successful   Failing     Failing
    Size  Buffers     Buffers     Asides      Reads      Writes     Writes     CREADS      CWRITES     CREADS      CWRITES

     512      4           0        1,360        4         31          0          0           0           0           0
    1024      4           0          0          0          0          0          0           0           0           0
    2048      4           0          0          0          0          0          0           0           0           0
    4096      4           0          0          0          0          0          0           0           0           0
    8192      4           0          0          0          0          0          0           0           0           0
   12288      4           0          0          0          0          0          0           0           0           0
   16384      4           0          0          0          0          0          0           0           0           0
   20480      4           0          0          0          0          0          0           0           0           0
   24576      4           0          0          0          0          0          0           0           0           0
   28672      4           0          0          0          0          0          0           0           0           0
   32768      4           0          0          0          0          0          0           0           0           0
```

Figure 45. The LSR Pools Report

Table 26 (Page 1 of 5). Fields in the LSR Pools Report	
Field Heading	**Description**
LSR Pools	
Pool Number	is the identifying number of the LSR pool. This value may be in the range 1 through 8.
Time Created	is the time when this LSR pool was created. Source field: A08LBKCD
Maximum key length	is the length of the largest key of a VSAM data set which may use this LSR pool. Source field: A08BKKYL
Total number of strings	is the total number of VSAM strings defined for this LSR pool. Source field: A08BKSTN
Peak concurrently active strings	is the maximum number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently lower than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need. Source field: A08BKHAS
Total requests waited for strings	is the number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSR pool string resources. Source field: A08BKTSW
Peak requests waited for strings	is the highest number of requests that were queued at one time because all the strings in the pool were in use. Source field: A08BKHSW
Buffer Totals	
Data Buffers	is the number of data buffers specified for the LSR pool. Source field: A08TDBFN
Hiperspace Data Buffers	is the number of Hiperspace data buffers specified for the LSR pool. Source field: A08TDHBW
Successful lookasides	is the number of successful lookasides to data buffers for this LSR pool. Source field: A08TDBFF
Buffer reads	is the number of read I/Os to the data buffers for this LSR pool. Source field: A08TDFRD
User initiated writes	is the number of user-initiated buffer writes from the data buffers for this LSR pool. Source field: A08TDUIW
Non-user initiated writes	is the number of non-user-initiated buffer writes from the data buffers for this LSR pool. Source field: A08TDNUW
Successful Hiperspace CREADS	is the number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers. Source field: A08TDCRS

Table 26 (Page 1 of 5). Fields in the LSR Pools Report	
Field Heading	**Description**
Successful Hiperspace CWRITES	is the number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers. Source field: A08TDCWS
Failing Hiperspace CREADS	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Source field: A08TDCRF
Failing Hiperspace CWRITES	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the data to DASD. Source field: A08TDCWF
Index Buffers	is the number of index buffers specified for the LSR pool. Source field: A08TIBFN
Hiperspace Index Buffers	is the number of Hiperspace index buffers specified for the LSR pool. Source field: A08TIHBW
Successful look asides	is the number of successful lookasides to index buffers for this LSR pool. Source field: A08TIBFF
Buffer reads	is the number of read I/Os to the index buffers for this LSR pool. Source field: A08TIFRD
User initiated writes	is the number of user-initiated buffer writes from the index buffers for this LSR pool. Source field: A08TIUIW
Non-user initiated writes	is the number of non-user-initiated buffer writes from the index buffers for this LSR pool. Source field: A08TINUW
Successful Hiperspace CREADS	is the number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers. Source field: A08TICRS
Successful Hiperspace CWRITES	is the number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers. Source field: A08TICWS
Failing Hiperspace CREADS	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read index data from DASD. Source field: A08TICRF
Failing Hiperspace CWRITES	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the index data to DASD. Source field: A08TICWF
Data Buffer Statistics	
Size	is the size of the data buffers that are available to CICS. Source field: A08BKBSZ
Buffers	is the number of buffers of each size available to CICS. Source field: A08BKBFN

Table 26 (Page 1 of 5). Fields in the LSR Pools Report	
Field Heading	**Description**
Hiperspace Buffers	is the number of Hiperspace buffers specified for the pool. Source field: A08BKHBN
Look Asides	is the number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08BKBFF
Reads	is the number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08BKFRD
User Writes	is the number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08BKUIW
Non-User Writes	is the number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08BKNUW
Successful CREADS	is the number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers. Source field: A08BKCRS
Successful CWRITES	is the number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Source field: A08BKCWS
Failing CREADS	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Source field: A08BKCRF
Failing CWRITES	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the data to DASD. Source field: A08BKCWF
Index Buffer Statistics	

Table 26 (Page 2 of 5). Fields in the LSR Pools Report	
Field Heading	**Description**
Size	is the size of the index data buffers that are available to CICS. Source field: A08IKBSZ
Buffers	is the number of buffers of each size available to CICS. Source field: A08IKBFN
Hiperspace Buffers	is the number of Hiperspace buffers specified for the pool. Source field: A08IKHBN
Look Asides	is the number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested index record was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKBFF
Reads	is the number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKFRD
User Writes	is the number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKUIW
Non-User Writes	is the number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKNUW
Successful CREADS	is the number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers. Source field: A08IKCRS
Successful CWRITES	is the number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Source field: A08IKCWS
Failing CREADS	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Source field: A08IKCRF

Table 26 (Page 3 of 5). Fields in the LSR Pools Report	
Field Heading	**Description**
Failing CWRITES	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the data to DASD. Source field: A08IKCWF

Files report

Figure 46 on page 487 shows the format of the Files Report. The field headings and contents are described in Table 27 on page 487.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37         CICS/ESA Version 4 Release 1.0    PAGE    23

Files

            Access              LSR        Max        Read     Get Update    Browse       Add      Update      Delete      Remote
Filename    Method     Type     Pool     Num Recs   Requests    Requests    Requests   Requests   Requests    Requests    Deletes

ADMF        VSAM                0           0          0           0           0          0          0           0           0
ADMGIMP     VSAM                1           0          0           0           0          0          0           0           0
ADMUMSL     VSAM                1           0          0           0           0          0          0           0           0
ADMX        VSAM                1           0          0           0           0          0          0           0           0
DFHCMACD    VSAM                1           0          0           0           0          0          0           0           0
DFHCSD      VSAM                1           0          0           0           0          0          0           0           0
LOGFILE     VSAM       KSDS     2         512          0           0           0          0          0           0           0
LOGFILEA    VSAM                2           0          0           0           0          0          0           0           0
```

Figure 46. The Files Report

Table 27 (Page 1 of 2). Fields in the Files Report

Field Heading	Description
Files	
Filename	is the name of the file. Source field: EXEC CICS INQUIRE FILE
Access Method	indicates the access method for this file. Source field: EXEC CICS INQUIRE FILE ACCESSMETHOD
Type	indicates how the records are organized in the data set that corresponds to this file. Source field: EXEC CICS INQUIRE FILE TYPE
LSR Pool	The identity of the LSR pool defined for this file. '0' means that it is not defined in an LSR pool. Source field: EXEC CICS INQUIRE FILE LSRPOOLID
Max Num Recs	is the maximum number of records that the data table for this file can hold. Source field: EXEC CICS INQUIRE FILE MAXNUMRECS
Read Requests	is the number of GET requests attempted against this file. Source field: A17DSRD
Get Update Requests	is the number of GET UPDATE requests attempted against this file. Source field: A17DSGU
Browse Requests	is the number of GETNEXT and GETPREV requests attempted against this file. Source field: A17DSBR
Add Requests	is the number of PUT requests attempted against this file. Source field: A17DSWRA
Update Requests	is the number of PUT UPDATE requests attempted against this file. Source field: A17DSWRU

Table 27 (Page 1 of 2). Fields in the Files Report	
Field Heading	**Description**
Delete Requests	is the number of DELETE requests attempted against this local file. Source field: A17DSDEL
Remote Deletes	is the number of DELETE requests for a VSAM file in a remote system. In systems connected by a CICS intercommunication (MRO or ISC) link, the statistics recorded for the remote files are a subset of those recorded for the files on the local system. Source field: A17RMDEL

Data Tables Reports

Figure 47 on page 489 shows the format of the Data Tables Requests Report. The field headings and contents are described in Table 28 on page 489.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37        CICS/ESA Version 4 Release 1.0    PAGE    25

Data Tables - Requests

                Data       Successful   Records    Adds via    Adds via       Adds       Adds      Rewrite      Delete       Read
Filename    Table Type       Reads     Not Found     Read         API        Rejected     Full     Requests     Requests     Retries
-------------------------------------------------------------------------------------------------------------------------------

ACCTFILE CICS/                   0          0          0           0           0          0           0            0           0
LOGFILE  CICS/TABLE              0          0          0           2           0          0           0            0           0
```

Figure 47. The Data Tables Requests Report

Table 28 (Page 1 of 2). Fields in the Data Tables Requests Report

Field Heading	Description
Data Tables - Requests	
Filename	is the name of the file. Source field: EXEC CICS INQUIRE FILE
Data Table Type	indicates whether this file represents a data table. Source field: EXEC CICS INQUIRE FILE TABLE
Successful Reads	is the number of attempts to retrieve records from the table. Source field: A17DTRDS
Records Not Found	is the number of times API READ requests were directed to the source data set because the record was not found in the table. Source field: A17DTRNF
Adds via Read	is the number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress. Source field: A17DTAVR
Adds via API	is the number of attempts to add records to the table as a result of WRITE requests. Source field: A17DTADS
Adds Rejected	is the number of records CICS attempted to add to the table which were rejected by the global user exit. Source field: A17DTARJ
Adds Full	is the number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified. Source field: A17DTATF
Rewrite Requests	is the number of attempts to update records in the table as a result of REWRITE requests. Source field: A17DTRWS

Table 28 (Page 1 of 2). Fields in the Data Tables Requests Report	
Field Heading	**Description**
Delete Requests	is the number of attempts to delete records from the table as a result of DELETE requests. Source field: A17DTDLS
Read Retries	is the total number of read retries, that is the number of times reads in an AOR had to be retried because the FOR changed the table during the read. Source field: A17DTRRS

Figure 48 on page 490 shows the format of the Data Tables Storage Report. The field headings and contents are described in Table 29 on page 490.

```
Applid IYAHZCH2   Sysid CI41   Jobname CI07CJB1   Date 10/06/93   Time 08:37:37         CICS/ESA Version 4 Release 1.0    PAGE    26

Data Tables - Storage

                                        <------- Total ------>  <------ Entries ----->  <------- Index ------>  <------- Data ------->
                  Current      Peak      Storage    Storage      Storage    Storage      Storage    Storage      Storage    Storage
Filename Type     Records     Records    Allocated   In-Use      Allocated   In-Use      Allocated   In-Use      Allocated   In-Use
_____________________________________________________________________________________________________________________________
ACCTFILE CICS           0           0           0           0           0           0           0           0           0           0
LOGFILE  CICS           0          78       1,033           0           0           0           0           0           0           0
```

Figure 48. The Data Tables Storage Report

Table 29 (Page 1 of 2). Fields in the Data Tables Storage Report	
Field Heading	**Description**
Data Tables - Storage	
Filename	is the name of the file. Source field: EXEC CICS INQUIRE FILE
Type	is the type of data table, CICS-maintened or user-maintained. Source field: EXEC CICS INQUIRE FILE TABLE
Current Records	is the current number of records in the data table. Source field: A17DTSIZ
Peak Records	is the peak number of records in the data table. Source field: A17DTSHI
Total - Storage Allocated	is the total amount of storage (kilobytes) in allocated for the data table. Source field: A17DTALT
Total - Storage In-Use	is the total amount of storage (kilobytes) in use for the data table. Source field: A17DTUST
Entries - Storage Allocated	is the total amount of storage (kilobytes) allocated for the record entry blocks. Source field: A17DTALE
Entries - Storage In-Use	is the total amount of storage (kilobytes) in use for the record entry blocks. Source field: A17DTUSE
Index - Storage Allocated	is the total amount of storage (kilobytes) allocated for the index. Source field: A17DTALI

Table 29 (Page 1 of 2). Fields in the Data Tables Storage Report	
Field Heading	**Description**
Index - Storage In-Use	is the total amount of storage (kilobytes) in use for the index. Source field: A17DTUSI
Data - Storage Allocated	is the total amount of storage (kilobytes) allocated for the record data. Source field: A17DTALD
Data - Storage In-Use	is the total amount of storage (kilobytes) in use for the record data. Source field: A17DTUSD

Appendix C. MVS and CICS virtual storage

Diagnosis, Modification or Tuning Information

Important note

This appendix provides some data relating to other products installed with CICS. This information was valid when this book was written, but no guarantee can be given that the information will stay unchanged, either for other products or for CICS itself. You should always check the information with your local IBM Systems Center.

This appendix describes:

- Major elements of “MVS storage” on page 493
- Major elements of “The CICS private area” on page 496
- Contents of “The dynamic storage areas” on page 503.

Most of the CICS storage areas are moved above the line, and it is necessary to have some detailed knowledge of the components that make up the total address space in order to determine what is really required.

Furthermore, it is important to understand the implications of the value of MXT (maximum tasks) on the storage use within the CICS address space. The value chosen for MXT and the size specified for the DSA limit and the EDSA limit. For a given region size, a high MXT value reduces the storage available for other users in the dynamic storage areas.

MVS storage

There are four major elements of virtual storage within MVS. Each storage area is duplicated above 16MB.

- The common area below 16MB
- The private area below 16MB
- The extended common area above 16MB
- The extended private area above 16MB.

The MVS common area

The common areas contain the following elements:

- MVS/ESA nucleus and extended nucleus
- System queue area (SQA and ESQA)
- Link pack areas (PLPA, MLPA, and CLPA)
- Common system area (CSA and ECSA)
- Prefixed storage area (PSA).

All the elements of the common area above are duplicated above 16MB line with the exception of the PSA.

MVS nucleus and MVS extended nucleus

This is a static area containing the nucleus load module and extension to the nucleus. Although its size is variable depending on an installation's configuration, it cannot change without a re-IPL of MVS. The nucleus area below the 16MB line does not include page frame table entries, and the size of the nucleus area is rounded up to a 4KB boundary. In addition, the nucleus area is positioned at the top of the 16MB map while the extended nucleus is positioned just above 16MB.

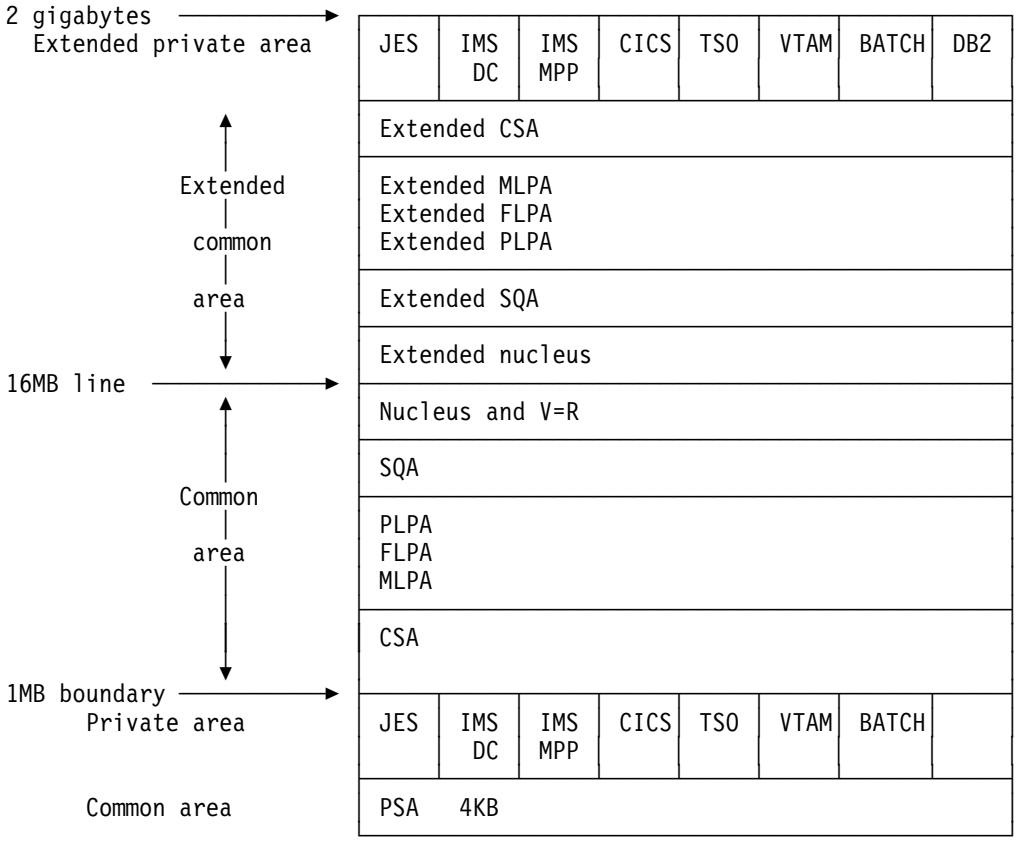


Figure 49. Virtual storage map for MVS/ESA

System queue area (SQA) and extended system queue area (ESQA)

This area contains tables and queues relating to the entire system. Its contents are highly dependent on configuration and job requirements at an installation. The total amount of virtual storage, number of private virtual storage address spaces, and size of the installation performance specification table are some of the factors that affect the system's use of SQA. The size of the initial allocation of SQA is rounded up to a 64KB boundary, though SQA may expand into the common system area (CSA) in increments of 4KB.

If the SQA is overallocated, the virtual storage is permanently wasted. If it is underallocated, it expands into CSA, if required. In a storage constrained system, it is better to be slightly underallocated. This can be determined by looking at the amount of free storage.

If the extended SQA is underallocated, it expands into the extended CSA. When both the extended SQA and extended CSA are used up, the system allocates storage from SQA and CSA below the 16MB line. The allocation of this storage could eventually lead to a system failure, so it is better to overallocate extended SQA and extended CSA.

Link pack area (LPA) and extended link pack area (ELPA)

The link pack area (LPA) contains all the common reentrant modules shared by the system, and exists to provide:

- Economy of real storage by sharing one copy of the modules
- Protection: LPA code cannot be overwritten even by key 0 programs
- Reduced pathlength, because modules can be branched to.

It has been established that a 2MB LPA is sufficient for MVS when using CICS with MRO or ISC, that is, the size of an unmodified LPA as shipped by IBM. If it is larger, there are load modules in the LPA that may be of no benefit to CICS. There may be SORT, COBOL, ISPF, and other modules that are benefiting batch and TSO users. You have to evaluate if the benefits you are getting are worth the virtual storage that they use. If modules are removed, be sure to determine if the regions they run in need to be increased in size to accommodate them.

The pageable link pack area (PLPA) contains supervisor call routines (SVCs), access methods, and other read-only system programs along with read-only re-enterable user programs selected by an installation to be shared among users of the system. Optional functions or devices selected by an installation during system generation add additional modules to the PLPA.

The modified link pack area (MLPA) contains modules that are an extension to the PLPA. The MLPA may be changed at IPL without requiring the create link pack area (CLPA) option at IPL to change modules in the PLPA.

MVS/ESA common system area (CSA) and extended common system area (ECSA)

These areas contain pageable system data areas that are addressable by all active virtual storage address spaces. They contain, for example, buffers or executable modules for IMS/ESA, ACF/VTAM, JES3, and so on. Also present are control blocks used to define subsystems and provide working storage for such areas as TSO input/output control (TIOC), event notification facility (ENF), message processing facility (MPF), and so on. As system configuration and activity increase, the storage requirements also increase.

CICS uses the ECSA only if IMS/ESA or MRO is used. Even in this case, this use is only for control blocks and not for data transfer. If cross-memory facilities are being used, the ECSA usage is limited to 20 bytes per session and 1KB per address space participating in MRO. The amount of storage used by CICS MRO is given in the DFHIR3794 message issued to the CSMT destination at termination.

For static systems, the amount of unallocated CSA should be around 10% of the total allocated CSA; for dynamic systems, a value of 20% is desirable. Unlike the SQA, if CSA is depleted there is no place for it to expand into and a re-IPL can very possibly be required.

Prefixed storage area (PSA)

The PSA contains processor-dependent status information such as program status words (PSWs). There is one PSA per processor; however, all of them map to virtual storage locations 0–4KB as seen by that particular processor. MVS/ESA treats this as a separate area; there is no PSA in the extended common area.

Private area and extended private area

The private area contains:

- A local system queue area (LSQA)
- A scheduler work area (SWA)
- Subpools 229 and 230 (the requestor protect key area)
- A 16KB system region area (used by the initiator)
- A private user region for running programs and storing data.

Except for the 16KB system region area, each storage area in the private area has a counterpart in the extended private area.

The portion of the user's private area within each virtual address space that is available to the user's application program, is referred to as its **region**. The private area user region may be any size up to the size of the entire private area (from the top end of the PSA to the beginning, or bottom end, of the CSA) ***minus*** the size of LSQA, SWA, subpools 229 and 230, and the system region: for example, 220KB. (It is recommended that the region be 250KB less to allow for RTM processing.)

The segment sizes are one megabyte, therefore CSA is rounded up to the nearest megabyte. The private area is in increments of one megabyte. For more information, see "The CICS private area" on page 496.

The CICS private area

The CICS private area has both static and dynamic storage requirements. The static areas are set at initialization time and do not vary over the execution of that address space. The dynamic areas increase or decrease their allocations as the needs of the address space vary, such as when data sets are opened and closed, and VTAM inbound messages being queued.

This section describes the major components of the CICS address space. In CICS/ESA 4.1 there are eight dynamic storage areas. They are:

The user DSA (UDSA)
: The user-key storage area for all user-key task-lifetime storage below the 16MB boundary.

The read-only DSA (RDSA)
: The key-0 storage area for all reentrant programs and tables below the 16MB boundary.

The shared DSA (SDSA)
: The user-key storage area for any non-reentrant user-key RMODE(24) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage below the 16MB boundary with the SHARED option.

The CICS DSA (CDSA)
The CICS-key storage area for all non-reentrant CICS-key RMODE(24) programs, all CICS-key task-lifetime storage below the 16MB boundary, and for CICS control control blocks that reside below the 16MB boundary.

The extended user DSA (EUDSA)
The user-key storage area for all user-key task-lifetime storage above the 16MB boundary.

The extended read-only DSA (ERDSA)
The key-0 storage area for all reentrant programs and tables above the 16MB boundary.

The extended shared DSA (ESDSA)
The user-key storage area for any non-reentrant user-key RMODE(ANY) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage above the 16MB boundary with the SHARED option.

The extended CICS DSA (ECDSA).
The CICS-key storage area for all non-reentrant CICS-key RMODE(ANY) programs, all CICS-key task-lifetime storage above the 16MB boundary, and CICS control blocks that reside above the 16MB boundary.

Figure 50 on page 498 shows an outline of the areas involved in the private area. The three main areas are HPA, MVS storage, and the CICS region. The exact location of the free and allocated storage may vary depending on the activity and the sequence of the GETMAIN/FREEMAIN requests.

Additional MVS storage may be required by CICS for kernel stack segments for CICS system tasks—this is the CICS kernel.

Note: The CICS extended private area is conceptually the same as the CICS private area except that there is no system region. All the other areas have equivalent areas above the 16MB line.

High private area

This area consists of four areas:

- LSQA
- SWA
- Subpool 229
- Subpool 230.

The area at the high end of the address space is not specifically used by CICS, but contains information and control blocks that are needed by the operating system to support the region and its requirements.

The usual size of the high private area varies with the number of job control statements, messages to the system log, and number of opened data sets.

The total space used in this area is reported in the IEF374I message in the field labeled “SYS=nnnnK” at jobstep termination. There is a second “SYS=nnnnK” which is issued which refers to the high private area above 16MB. This information is also reported in the sample statistics program, DFH0STAT.

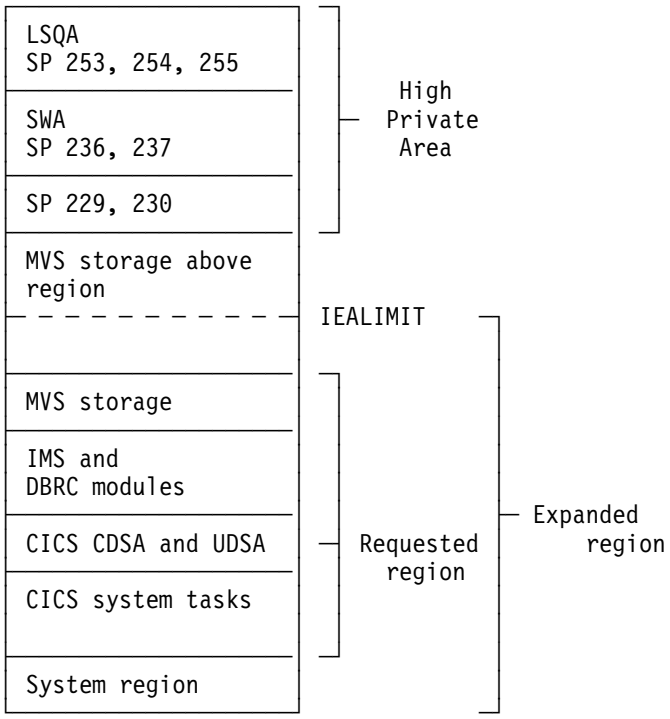


Figure 50. CICS private area immediately after system initialization

Very little can be done to reduce the size of this area, with the possible exception of subpool 229. This is where VTAM stores inbound messages when CICS does not have an open receive issued to VTAM. The best way to determine if this is happening is to use CICS statistics (see "VTAM statistics: Global statistics" on page 438) obtained following CICS shutdown. Compare the maximum number of RPLs posted, which is found in the shutdown statistics, with the RAPOOL value in the SIT. If they are equal, there is a very good chance that subpool 229 is being used to stage messages, and the RAPOOL value should be increased.

The way in which the storage within the high private area is used can cause an S80A abend in some situations. There are at least two considerations:

1. **The use of MVS subpools 229 and 230 by access methods such as VTAM**: VTAM and VSAM may find insufficient storage for a request for subpools 229 and 230. Their requests are conditional and so should not cause an S80A abend of the job step (for example, CICS).
2. **The MVS operating system itself, relative to use of LSQA and SWA storage during job-step initiation**: The MVS initiator's use of LSQA and SWA storage may vary, depending on whether CICS was started using an MVS START command or started as a job step as part of already existing initiator and address space. Starting CICS with an MVS START command is better for minimizing fragmentation within the space above the region boundary. If CICS is a job step initiated in a previously started initiator's address space, the manner in which LSQA and SWA storage is allocated may reduce the apparently available virtual storage because of increased fragmentation.

Storage above the region boundary must be available for use by the MVS initiator (LSQA and SWA) and the access method (subpools 229 and 230).

Consider initiating CICS using an MVS START command, to minimize fragmentation of the space above your specified region size. This may avoid S80A abends by more effective use of the available storage.

Your choice of sizes for the MVS nucleus, MVS common system area, and CICS region influences the amount of storage available for LSQA, SWA, and subpools 229 and 230. It is unlikely that the sizes and boundaries for the MVS nucleus and common system area can easily be changed. To create more space for the LSQA, SWA, and subpools 229 and 230, you may need to ***decrease*** the region size.

Local system queue area (LSQA)

This area generally contains the control blocks for storage and contents supervision. Depending on the release level of the operating system, it may contain subpools 233, 234, 235, 253, 254, and 255.

The total size of LSQA is difficult to calculate because it depends on the number of loaded programs, tasks, and the number and size of the other subpools in the address space. As a guideline, the LSQA area usually runs between 40KB and 170KB depending on the complexity of the rest of the CICS address space.

The storage control blocks define the storage subpools within the private area, describing the free and allocated areas within those subpools. They may consist of such things as subpool queue elements (SPQEs), descriptor queue elements (DQEs), and free queue elements (FQEs).

The contents management control blocks define the tasks and programs within the address space such as task control blocks (TCBs), the various forms of request blocks (RBs), contents directory elements (CDEs), and many more.

CICS DBCTL requires LSQA storage for DBCTL threads. Allow 9KB for every DBCTL thread, up to the MAXTHRED value.

Scheduler work area (SWA)

This area is made up of subpools 236 and 237, which contain information about the job and step itself. Almost anything that appears in the job stream for the step creates some kind of control block here.

Generally, this area can be considered to increase with an increase in the number of DD statements. The distribution of storage in subpools 236 and 237 varies with the operating system release and whether dynamic allocation is used. The total amount of storage in these subpools is around 100KB to 150KB to start with, and it increases by about 1KB to 1.5KB per allocated data set.

A subset of SWA control blocks can, optionally, reside above 16MB. JES2 and JES3 have parameters that control this. If this needs to be done on an individual job basis, the SMF exit, IEFUJV, can be used.

Subpool 229

This subpool is used primarily for the staging of messages. JES uses this area for messages to be printed on the system log and JCL messages as well as SYSIN/SYSOUT buffers. Generally, a value of 40 to 100 KB is acceptable, depending on the number of SYSIN and SYSOUT data sets and the number of messages in the system log.

Subpool 230

This subpool is used by VTAM for inbound message assembly for segmented messages. Data management keeps data extent blocks (DEBs) here for any opened data set.

Generally, the size of subpool 230 increases as the number of opened data sets increases. Starting with an initial value of 40KB to 50KB, allow 300 to 400 per opened data set.

CICS DBCTL requires subpool 230 storage for DBCTL threads. Allow 3KB for every DBCTL thread, up to the MAXTHRED value.

MVS storage above region

This is the storage that is left between the top of the region and the bottom of the high private area. We recommend that 200KB to 300KB of free storage be maintained to allow for use by the termination routines in the case of an abend.

If this free storage is not enough for recovery termination management (RTM) processing, the address space may be terminated with a S40D abend that produces no dump.

This area can be very dynamic. As the high private area grows, it extends down into this area, and the CICS region may extend upward into this area up to the value specified in IEALIMIT.

The CICS region

The total storage for a CICS region is reported by the IEF374I message in the "VIRT=nnnK" portion for most address spaces. The equivalent message above the 16MB line is "EXT=nnnK". The sample statistics program, DFH0STAT, produces a report with this information. CICS formatted dumps and some specialized monitors may be useful to determine the sizes of the various components mentioned below. CICS statistics reports contain useful information about the subpools of the dynamic storage areas.

CICS virtual storage

CICS virtual storage requirements can be divided into:

- ***MVS Storage***: storage available to the operating system to perform region related services.
- ***Dynamic storage area***: the requirements of CICS, access methods, and applications.

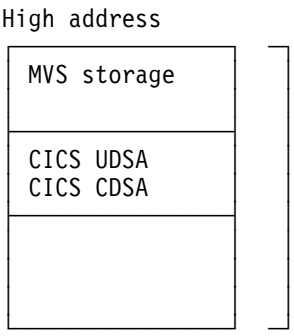


Figure 51. CICS region layout

MVS storage

The part of the CICS address space called MVS storage is the storage available to the MVS operating system to perform region-related services in response to an operating system macro or SVC issued by the region. For example, operating system components such as VSAM, DL/I, or DB2, issue MVS GETMAIN requests to obtain storage in which to build control blocks and these requests are met from MVS storage.

This is the amount of storage that remains after the dynamic storage areas and other CICS storage requirements have been met. The size of this area depends on MVS GETMAIN requirements during the execution of CICS. Opening files is the major contributor to usage of this area.

MVS storage is used to contain control blocks and data areas needed to open data sets or do other operating system functions, and program modules for the access method routines not already resident in the LPA, and shared routines for the COBOL and PL/I programs.

The VSAM buffers and most of the VSAM file control blocks reside above the 16MB line.

The VSAM buffers may be for CICS data sets defined as using local shared resources (LSR) (that is, the default) or nonshared resources (NSR).

The VSAM LSR pool is built dynamically above the 16MB line when the first file specified as using it is opened, and deleted when the last file using it is closed.

Every opened data set requires some amount of storage in this area for such items as input/output blocks (IOBs) and channel programs.

Files that are defined as data tables use storage above 16MB for records that are included in the table, and for the structures which allow them to be accessed.

QSAM files require some storage in this area. Transient data uses a separate buffer pool above the 16MB line for each type of transient data queue. Storage is obtained from the buffer pool for queue entries as they are added to the destination control table. Transient data also uses a buffer pool below the 16MB line where sections of extrapartition DCTEs are copied for use by QSAM, when an extrapartition queue is being opened or closed.

CICS DBCTL uses DBCTL threads. DBCTL threads are specified in the CICS address space but they have storage requirements in the high private area of the CICS address space.

If DB2 is used by the system, MVS storage is allocated for each DB2 thread.

The physical placement of the MVS storage may be anywhere within the region, and may sometimes be above the CICS region. The region may expand into this MVS storage area up to the IEALIMIT set by the installation or up to the default value (see the *MVS/ESA Programming: Callable Services for High-Level Languages*). This expansion occurs when operating system GETMAIN requests are issued, the MVS storage within the region is exhausted, and the requests are met from the MVS storage area above the region.

When both the MVS storage areas are exhausted, the GETMAIN request fails, causing abends or a bad return code if it is a conditional request.

The amount of MVS storage must be enough to satisfy the requests for storage during the entire execution of the CICS region. You must use caution here; you never want to run out of MVS Storage but you do not want to overallocate it either.

The size of MVS storage is the storage which remains in the region after allowing for the storage required for the dynamic storage areas, the kernel storage areas, and the IMS/VS and DBRC module storage. The specification of OSCOR storage in CICS/MVS Version 2 and earlier has been replaced with the specification of the DSA sizes in CICS/ESA Version 3. It is important to specify the correct DSA sizes so that the required amount of MVS storage is available in the region.

Because of the dynamic nature of a CICS system, the demands on MVS storage varies through the day, that is, as the number of tasks increases or data sets are opened and closed. Also, because of this dynamic use of MVS storage, fragmentation occurs, and additional storage must be allocated to compensate for this.

VSAM storage

The VSAM buffers and most of the VSAM file control blocks reside above the 16MB line.

The VSAM buffers may be for CICS data sets defined as using local shared resources (LSR) (the default) or nonshared resources (NSR). If DL/I data sets are used, the buffers for these data sets always use an LSR pool.

The VSAM LSR pool is built dynamically above the 16MB line when the first file specified as using it is opened, and deleted when the last file using it is closed.

Other file storage

Every opened data set requires some amount of storage in this area for such items as input/output blocks (IOBs) and channel programs.

The dynamic storage areas

The dynamic storage areas are used to supply CICS tasks with the storage needed to execute your transactions. From the storage size that you specify on the DSALIM and the EDSALIM parameters, CICS allocates the dynamic storage areas above and below the line respectively.

Too small a dynamic storage area results in increased program compression or, more seriously, SOS (short on storage) conditions, or even storage deadlock abends when program compression is not sufficient.

DSAs consist of one or more extents. An extent below the line is 256KB and above the line, 1MB (except UDSA with TRANISO active, when the extent is 1M).

CICS GETMAIN requests for dynamic storage are satisfied from one of the following: the CDSA, RDSA, SDSA, ESDSA, UDSA, ECDSA, or the ERDSA during normal execution. The sizes of the dynamic storage areas are defined at CICS initialization, but the use of storage within them is very dynamic.

The dynamic storage areas consist of a whole number of virtual storage pages allocated from a number of MVS storage subpools. CICS uses about 180 storage subpools, which are located in the dynamic storage areas. For a list of the subpools see the tables on pages 505 through 515. Each dynamic storage area has it own “storage cushion” These subpools (including the cushion) are dynamically acquired, as needed, a page at a time, from within the dynamic storage area.

The dynamic storage areas are essential for CICS operation. Their requirements depend on many variables, because of the number of subpools. The major contributors to the requirements are program working storage and the type and number of program and terminal definitions. The storage used by individual subpools can be determined by examining the CICS domain subpool statistics (storage manager statistics).

If you have exhausted the tuning possibilities of MVS/ESA and other tuning possibilities outside CICS, and the dynamic storage areas limits have been set as large as possible within CICS, and you are still encountering virtual storage constraint below 16MB, you may have to revise the use of options such as MXT and making programs resident, to keep down the overall storage requirement. This may limit task throughput. If you foresee this problem on an MVS system, you should consider ways of dividing your CICS system, possibly by use of facilities such as CICS multiregion operation (MRO), described in the *CICS/ESA Intercommunication Guide*. You can also reduce storage constraint below 16MB by using programs which run above 16MB.

In systems with a moderate proportion of loadable programs, program compression is an indicator of pressure on virtual storage. This can be determined by examining the CICS storage manager statistics which report the number of times that CICS went short on storage.

If the dynamic storage areas limits are too small, CICS performance is degraded. The system may periodically enter a short-on-storage condition, during which it curtails system activity until it can recover enough storage to resume normal operations.

However, resist the temptation to make the dynamic storage area limit as large as possible (which you could do by specifying the maximum allowable region size). If you do this, it can remove any warning of a shortage of virtual storage until the problem becomes intractable.

The dynamic storage areas limits should be as large as possible after due consideration of other areas, especially the MVS storage area above 16MB.

CICS subpools

This section describes briefly the main features of the subpools. They are found in each of the dynamic storage areas. Most of the subpools are placed above the 16MB line. Those subpools that are found below the 16MB line, in the CDSA, SDSA, RDSA, and UDSA, need to be more carefully monitored due to the limited space available. Individual subpools may be static or dynamic. Some contain static CICS storage which cannot be tuned. All the subpools are rounded up to a multiple of 4KB in storage size and this rounding factor should be included in any subpool sizing or evaluation of storage size changes due to tuning or other changes. CICS statistics contain useful information about the size and use of the dynamic storage area subpools. The CICS subpools in the dynamic storage areas may be grouped and described according to the major factor affecting their use.

Application design
: The use of CICS facilities such as program LINK, SHARED storage GETMAINs, the type of file requests, use of temporary storage, application program attributes (resident or dynamic), or the number of concurrent DBCTL, or DB2, requests affect storage requirements.

Number of files definitions
: These subpools may only be tuned by reducing the number of file definitions, or by using MRO.

The use of DBCTL, or DB2
: These subpools are only present if DBCTL or DB2 is used. The subpools may be tuned by reducing the number of threads, or by using maximum tasks (MXT) or transaction classes.

Nucleus and macro table storage
: It may be possible to reduce the macro table storage by reducing the number of macro definitions or by migrating selected macro-defined tables to RDO.

Number and type of terminal definitions
: The OPNDLIM system initialization parameter may also be tuned to limit storage use.

The following tables list the subpools according to their dynamic storage areas and their use. These tables do not include the storage subpools used by the Front End Programming Feature. See the *CICS/ESA Front End Programming Interface User's Guide* for more information.

CICS subpools in the CDSA

Table 30 (Page 1 of 2). CICS subpools in the CDSA

Subpool name	Description
DB2	contains the TIE blocks for RMI use, when invoked by the DB2 task-related user exit program, DSN2EXT1. The tie is 100 bytes long and appended to the tie is the local task work area for this task-related user exit which is for DSN2EXT1 2268 bytes long. This subpool is only present when DB2 is used. It may be tuned by limiting DB2 threads or using maximum tasks (MXT) or transaction classes.
DFHAPD24	is a general subpool for application domain. Storage below the line.
DFHTDG24	CXRE queue definitions and SDSCI are allocated from this subpool.
DFHTDSDS	contains real transient data SDCIs, each of which contains a DCB which resides below the line.
FC_DCB	contains the DCBs for BDAM files. Each file that is defined requires 104 bytes.
FCCBELOW	contains real VSWA and data buffers for pre-reads. Each VSWA requires 120 bytes of storage. The maximum number of data buffers for pre-reads is given by: (number of strings) x (maximum record length) x (number of files).
KESTK24	2KB below the line kernel stack. One per MXT plus one for every dynamic system task that is running.
KESTK24E	4KB below the line kernel stack extension. At least one of these for every ten tasks specified in the MXT limit.
LDNUC	contains the CICS nucleus and macro tables. The CICS nucleus is approximately 192KB and the size of the tables can be calculated. The FCT may be migrated to RDO if it is currently a macro table. See the *CICS/ESA Resource Definition Guide* for more information. **APAR PQ07674** MJO 26/08/98 Contains programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited REENTRANT and RMODE(24)
LDNRS	contains the CICS nucleus and macro tables. The CICS nucleus is approximately 192KB and the size of the tables can be calculated. The FCT may be migrated to RDO if it is currently a macro table. See the *CICS/ESA Resource Definition Guide* for more information. Contains programs defined EXECKEY(CICS) which are RESIDENT, that were link edited REENTRANT and RMODE(24)
SMCONTRL	satisfies GETMAINs for control class storage.
SMSHARED	contains shared storage below the 16MB line, for example RMI global work areas, EDF blocks for the life of the transaction being monitored, and other control blocks.
SMSHRC24	is used for many control blocks of SHARED_CICS24 class storage.
SMTCA24	stores the TCA when the task data location option is set to BELOW.
SMTP24	holds line and terminal I/O areas which cannot be located above the 16MB line. The storage requirements depend on the amount of terminal and line traffic in the system. The subpool may be tuned by reducing the RAPOOL, RAMAX, TIOAL size, and number of MRO sessions.

Table 30 (Page 1 of 2). CICS subpools in the CDSA	
Subpool name	**Description**
SZSPFCAC	contains the FEPI VTAM ACB work areas.
ZCSETB24	contains application control buffers below the line.

CICS subpools in the SDSA

Table 31. CICS subpools in the SDSA	
Subpool name	**Description**
APECA	contains the event control areas.
DFHAPU24	is a general subpool for application domain storage below the line.
LDPGM	contains dynamically loaded application programs (RMODE (24)). The expected size of this subpool may be predicted from previous releases, and by taking LDPGMRO into account. The subpool size may be reduced by using 31-bit programs. Not reentrant.
LDRES	contains resident application programs (RMODE (24). The expected size of this subpool may be predicted from previous releases, and by taking LDRESRO into account. The subpool size may be reduced by using 31-bit programs. Not reentrant.
OSCOBOL	is used for the allocation of the COBOL merged load list (MLL) control block and its extents. It should never occupy more than its initial allocation of one page of storage.
SMSHRU24	is used for many control blocks of SHARED_USER24 class storage.
ZCTCTUA	contains the TCTTE user area. It can be located in of the following DSAs: CDSA, SDSA, ECDSA, or ESDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANY|BELOW and the system initialization parameter, TCTUAKEY=CICS|USER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the *CICS/ESA Resource Definition Guide* for more information about the terminal definition.

CICS subpools in the RDSA

Table 32. CICS subpools in the RDSA	
Subpool name	**Description**
LDNUCRO	contains programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited REENTRANT and RMODE(24).
LDENRS	contains programs defined EXECKEY(CICS) which are RESIDENT, that were link edited REENTRANT and RMODE(24).
LDPGMRO	contains programs defined EXECKEY(USER) which are not RESIDENT, that were link edited RMODE(24) and REENTRANT.
LDRESRO	contains programs defined EXECKEY(USER) and RESIDENT and were link edited REENTRANT and were link edited REENTRANT and RMODE.

CICS subpools in the ECDSA

Table 33 (Page 1 of 8). CICS subpools in the ECDSA

Subpool name	Description
AITM_TAE	is the autoinstall terminal model (AITM) table entry subpool (DFHAITDS).
AP_AFCTE	contains the application part of the file control table (FCT) for all local and remote files that are defined. Each VSAM file requires 48 bytes of storage and each remote file requires 64 bytes.
AP_TCA31	contains the application part of the TCA table
AP_TXDEX	contains the application part of the TXD table
APBMS	contains storage use by BMS.
APCOMM31	contains COMMAREAs. The storage requirement depends on the size of COMMAREA specified and the number of concurrent users of the application.
APEISTAC	contains storage used by the EXEC CICS interface.
APICAD31	contains storage for ICEs and AIDs above the line.
APLLASYS	contains system load list areas (LLA).
APRSAQCL	contains storage used by the EXEC CICS interface.
APURD	subpool contains URDs and nontask DWEs.
DBCTL	subpool contains the TIE blocks for RMI use, when invoked by the DBCTL task-related user exit program, DFHDBAT. The tie is 120 bytes long, and appended to the tie is the local task work area for this task-related user exit which is, for DFHDBAT, 668 bytes long. This subpool is present only when DBCTL is used. It may be tuned by limiting DBCTL threads or using maximum tasks (MXT) or transaction classes.
DCTE_EXT	contains all extrapartition queue definitions.
DCTE_IND	contains all indirect queue definitions.
DCTE_INT	contains all intrapartition queue definitions.
DCTE_REM	contains all remote queue definitions.
DDBROWSE	contains storage for directory manager browse request tokens.
DDGENRAL	contains directory manager control blocks general information.
DDS_PPT	contains storage for directory manager directory elements for the PPT table.
DDS_RTXD	contains storage for directory manager directory elements for the RTXD table.
DDS_TCL	contains storage for directory manager directory elements for the TCL table.
DDS_TPNM	contains storage for directory manager directory elements for the TPNM table.
DDS_TXD	contains storage for directory manager directory elements for the TXD table.
DDS_USD1	contains storage for directory manager directory elements for the USD1 table.

Table 33 (Page 1 of 8). CICS subpools in the ECDSA	
Subpool name	**Description**
DDS_USD2	contains storage for directory manager directory elements for the USD2.
DFHAPDAN	is a general subpool for application domain storage above the line.
DFHTDG31	contains transient data general storage and control blocks. The storage requirement depends on the number of buffers and strings, and on the control interval size specified.
DFHTDIOB	contains intrapartition transient data input/output buffers. The storage requirement is given by the control interval size of the intrapartition transient data set multiplied by the number of buffers.
DFHTDWCB	contains the transient data wait elements.
DL/I	subpool contains the TIE blocks for RMI use, when invoked by the EXEC DL/I task-related user exit program, DFHEDP. The tie is 120 bytes long, and appended to the tie is the local task work area for this task-related user exit which is, for DFHEDP, 4 bytes long. This subpool is only present when EXEC DL/I is used. It may be tuned by limiting DBCTL threads or using maximum tasks (MXT) or transaction classes.
DMSUBPOL	is the domain manager subpool for general usage.
DSBROWSE	contains storage for dispatcher browse request tokens.
FC_ABOVE	contains real VSWA and data buffers for pre-reads. Each VSWA requires 120 bytes of storage. The maximum number of data buffers for pre-reads is given by: (number of strings) x (maximum record length) x (number of files)
FC_ACB	contains ACBs for VSAM files. There is one ACB, of 80 bytes, per VSAM file.
FC_BDAM	BDAM file control blocks. Each BDAM file requires 96 bytes of storage.
FC_DSNAM	contains data set name blocks. Each file requires a data set name block which uses 120 bytes of storage.
FC_FFLE	contains the fast file elements (FFLEs). A FFLE is built each time a transaction references a file name that has not previously been referenced by that transaction. The FFLE is retained until the end of the task. There is a free chain of FFLEs not currently in use.
FC_FRAB	contains file request anchor blocks (FRABs). There is one FRAB for each transaction that has issued a file control request. The FRAB is retained until the end of the task. There is a free chain of FRABs not currently in use.
FC_FRTE	contains file request thread elements (FRTE). There is one FRTE for each active file control request per task. A file control request has a FRTE if: • It has not yet terminated its VSAM thread. For example, a browse that has not yet issued an ENDBR. • It has updated a recoverable file and there has not yet been a syncpoint. • It is holding READ-SET storage that must be freed in future. There is a free chain of FRTEs not currently in use.

Table 33 (Page 1 of 8). CICS subpools in the ECDSA

Subpool name	Description
FC_SHRCT	contains file control SHRCTL blocks. There are eight of these and each describes a VSAM LSR pool.
FC_VSAM	contains the file control table (FCT) entries for VSAM files.
FCB_C1K	contains file control buffers of length 1KB. They are used by file control requests that are made against files whose maximum record length is between 512 bytes+1 byte up to 1KB.
FCB_C12K	contains file control buffers of length 12KB. They are used by file control requests that are made against files whose maximum record length is between 8KB+1 byte up to 12KB.
FCB_C16K	contains file control buffers of length 16KB. They are used by file control requests that are made against files whose maximum record length is between 12KB+1 byte up to 16KB.
FCB_C2K	contains file control buffers of length 2KB. They are used by file control requests that are made against files whose maximum record length is between 1KB+ 1 byte up to 2KB.
FCB_C20K	contains file control buffers of length 20KB. They are used by file control requests that are made against files whose maximum record length is between 16KB+1 byte up to 20KB.
FCB_C24K	contains file control buffers of length 24KB. They are used by file control requests that are made against files whose maximum record length is between 20KB+1 byte up to 24KB.
FCB256	contains file control buffers of length 256 bytes. They are used by file control requests that are made against files whose maximum record length is less than or equal to 256 bytes.
FCB_C28K	contains file control buffers of length 28KB. They are used by file control requests that are made against files whose maximum record length is between 24KB+1 byte up to 28KB.
FCB_C32K	contains file control buffers of length 32KB. They are used by file control requests that are made against files whose maximum record length is between 28KB+1 byte up to 32KB.
FCB_C4K	contains file control buffers of length 4KB. They are used by file control requests that are made against files whose maximum record length is between 2KB+1 byte up to 4KB.
FCB_C512	contains file control buffers of length 512 bytes. They are used by file control requests that are made against files whose maximum record length is between 256 bytes+1 byte up to 512 bytes.
FCB_C8K	contains file control buffers of length 8KB. They are used by file control requests that are made against files whose maximum record length is between 4KB+1 byte up to 8KB.
KESTK31	24KB above the line kernel stack. One per MXT plus one for every dynamic system task that is running.
KESTK31E	4KB above the line kernel stack extensions. At least one for every ten tasks specified in the MXT limit.
KETASK	kernel task entries.
L2OFL2BL	log manager domain - logger block entries.
L2OFL2BS	log manager domain - logger browseable stream objects.

Table 33 (Page 2 of 8). CICS subpools in the ECDSA

Subpool name	Description
L2OFL2CH	log manager domain - logger chain objects.
L2OFL2SR	log manager domain - logger stream objects.
LD_APES	loader domain - active program elements.
LD_CNTRL	loader domain - general control information.
LD_CSECT	loader domain - CSECT list storage.
LDENUC	contains the extended CICS nucleus, and 31-bit macro tables. The extended CICS nucleus is approximately 50KB. **APAR PQ07674** MJO 26/08/98 Contains programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited REENTRANT and RMODE(24).
LDNRSRO	contains the extended CICS nucleus, and 31-bit macro tables. The extended CICS nucleus is approximately 50KB. Contains programs defined EXECKEY(CICS) which are RESIDENT, that were link edited REENTRANT and RMODE(24).
LGBD	log manager domain - log stream name/journal name/journal model browse tokens.
LGGD	log manager domain - explicitly opened general logs.
LGGENRAL	general purpose subpool for log manager domain.
LGJI	log manager domain - journal name entries.
LGJMC	log manager domain - journal model resource entries.
LGSD	log manager domain - log stream data entries.
LGUOW	log manager domain - unit of work data entries.
LI_PLB	language interface - program language block. One is allocated for each program when control is first passed to it. **APAR PQ15774** MJO 18/08/98
MDTTABLE	MDT field attribute table for BMS maps sent through the CICS Web interface.
MN_CNTRL	contains monitoring control blocks - general information.
MN_MAES	contains monitoring control blocks. The storage requirement is 48 bytes per active task.
MN_TMAS	contains monitoring control blocks. The storage requirement is 472 bytes per active task.
MRO_QUEU	is used by the MRO work queue manager.
MROWORKE	is used by the MRO work queue manager elements.
PGGENRAL	general purpose program manager domain subpools.
PGHMRSA	program handle manager cobol register save areas.
PGHTB	program manager handle table block.
PGLLE	program manager load list elements.

Table 33 (Page 3 of 8). CICS subpools in the ECDSA	
Subpool name	**Description**
PGPGWE	program manager wait elements.
PGPPTE	program manager program definitions (PPTs).
PGPTA	program manager transaction-related information.
PR_TABLE	contains storage for PTEs from the PRT.
RM_TABLE	is used to quickcall the recovery manager lifetime control block.
SMSHRC31	is used for many control blocks of SHARED_CICS31 class storage.
SMTCA	stores the TCA when the task data location option is set to ANY.
SMTP	holds line and terminal I/O areas. The storage requirements depend on the amount of terminal and line traffic in the system. The subpool may be tuned by reducing the RAPOOL, RAMAX, TIOAL size, and number of MRO sessions.
STSUBPOL	is a statistics domain manager subpool.
SZSPFCCD	is the FEPI connection control subpool.
SZSPFCCM	is the FEPI common area subpool.
SZSPFCCV	is the FEPI conversation control subpool.
SZSPFCDS	is the FEPI device support subpool.
SZSPFCNB	is the FEPI node initialization block subpool.
SZSPFCND	is the FEPI node definition subpool.
SZSPFCPD	is the FEPI pool descriptor subpool.
SZSPFCPS	is the FEPI property descriptor subpool.
SZSPFCRP	is the FEPI request parameter list subpool.
SZSPFCRQ	is the FEPI requests subpool.
SZSPFCSR	is the FEPI surrogate subpool.
SZSPFCTD	is the FEPI target descriptor subpool.
SZSPFCWE	is the FEPI work element subpool.
SZSPVUDA	is the FEPI data areas subpool.
TD_TDCUB	contains all the transient data CI update control blocks.
TD_TDQUB	contains all the transient data queue update control blocks.
TD_TDUA	contains all the transient data UOW anchor control blocks.
TIA_POOL	is the timer domain anchor subpool.
TIQCPOOL	is the timer domain quickcell subpool.
TSBUFFRS	contains the temporary storage I/O buffers. The storage requirement is given by: (TS control interval size) x (number of TS buffers). The use of temporary storage by application programs affects the size of a number of subpools associated with temporary storage control blocks:
TSGENRAL	The amount of storage used by the TSGENRAL subpool depends on the number of buffers and strings and the control interval size defined for the temporary storage data set.

Table 33 (Page 4 of 8). CICS subpools in the ECDSA	
Subpool name	**Description**
TSMAIN	contains storage for temporary storage main storage. The subpool could be reduced by using auxiliary temporary storage.
TSMN0064	contains TS main items with lengths (including the header) less than or equal to 64.
TSMN0128	contains TS main items with lengths (including the header) less than or equal to 128.
TSMN0192	contains TS main items with lengths (including the header) less than or equal to 192.
TSMN0256	contains TS main items with lengths (including the header) less than or equal to 256.
TSMN0320	contains TS main items with lengths (including the header) less than or equal to 320.
TSMN0384	contains TS main items with lengths (including the header) less than or equal to 384.
TSMN0448	contains TS main items with lengths (including the header) less than or equal to 448.
TSMN0512	contains TS main items with lengths (including the header) less than or equal to 512.
TSTSS	contains TS section descriptors.
TSTSX	contains TS auxiliary item descriptors.
TSDTN	contains TS digital tree nodes.
TSQUEUE	contains TS queue descriptors.
TSBRB	contains TS browse blocks.
TSQAB	contains TS queue anchor blocks.
TSQOB	contains TS queue ownership blocks.
TSTSI	contains TS item descriptors.
TSQUB	contains TS queue update blocks.
TSICDATA	contains TS interval control elements.
TSW	contains TS wait queue elements.
UE_EPBPL	is the subpool for the user exit program block (EPB).
USGENRAL	is the general-purpose subpool for the user domain.
USIDTBL	contains the attach security userid table entries (LUITs). See Appendix D, “Performance data” on page 521 for more information.
USRTMQUE	contains queue elements for users waiting for USRDELAY. Each queue element is 16 bytes.
USUDB	contains user data blocks. The storage requirement is 128 bytes per unique user.
USXDPOOL	contains user domain transaction-related data. Each executing transaction requires 32 bytes.
VCTRLSUB	is used by volume control’s series definition table mechanism.
XMGENRAL	is the general-purpose subpool for the transaction manager

Table 33 (Page 5 of 8). CICS subpools in the ECDSA	
Subpool name	**Description**
XMLQEA	contains storage for large QEAs. The storage requirement depends on the number of concurrent ENQs for resources.
XMSQEA	contains storage for small QEAs. The storage requirement depends on the number of concurrent ENQs for resources.
XMTCLASS	contains the transaction manager tranclass definition.
XMTRANSN	transaction manager transactions. One for every transaction in the system.
XMTXDINS	transaction manager transaction definition.
XMTXDSTA	transaction manager transaction definition.
XMTXDTPN	contains the transaction manager transaction definition TPNAME storage.
XSGENRAL	is the general-purpose subpool for the security domain.
XSXMPOOL	contains security domain transaction-related data. Each executing transaction requires 56 bytes.
ZCBIMG	contains BIND images.
ZCBMSEXT	contains the BMS extensions for terminals. Subpool storage requirements are 48 bytes for each terminal, surrogate, ISC session, and console.
ZCBUF	contains the non-LU6.2 buffer list.
ZCCCE	contains the console control elements. Each console requires 48 bytes.
ZCGENERL	is the general-purpose subpool for terminal control.
ZCLUCBUF	contains the LU6.2 SEND and RECEIVE buffer list .
ZCLUCEXT	contains the LU6.2 extensions. The storage requirement is 224 bytes for each LU6.2 session.
ZCNIBD	contains the NIB descriptors. Each terminal, surrogate, ISC session, and system definition requires 96 bytes of storage.
ZCNIBISC	contains the expanded NIB and response during OPNDST/CLSDST for ISC. Each concurrent logon/logoff requires 448 bytes of storage. The maximum number of concurrent requests is limited by the number of sessions or OPNDLIM. The storage may be tuned by reducing the number of sessions or OPNDLIM. In systems running ACF/VTAM release 3.2 and later, the OPNDLIM parameter will be ignored because VTAM storage constraints are no longer a concern due to improved buffering techniques.
ZCNIBTRM	contains the expanded NIB during OPNDST/CLSDST for terminals. Each concurrent logon/logoff requires 192 bytes of storage. The maximum number of concurrent requests is limited by the number of terminals or OPNDLIM. The storage may be tuned by reducing the number of terminals or OPNDLIM.
ZCRAIA	contains the RECEIVE ANY I/O areas.
ZCRPL	contains the RPLs for active tasks. Each active task associated with a VTAM terminal requires 152 bytes.
ZCSETB	contains application control buffers above the line.

Table 33 (Page 6 of 8). CICS subpools in the ECDSA	
Subpool name	**Description**
ZCSKEL	contains the remote terminal entries. Each remote terminal definition requires 32 bytes of storage.
ZCSNEX	contain the TCTTE sign-on extensions. The storage requirement is 48 bytes for each terminal, surrogate, session, and console.
ZCTCME	contains the mode entries. Each mode entry requires 128 bytes of storage.
ZCTCSE	contains the system entries. Each system entry requires 192 bytes of storage.
ZCTCTTEL	contains the large terminal entries. 504 bytes of storage are required for every terminal, surrogate model, and ISC session defined.
ZCTCTTEM	contains the medium terminal entries. 400 bytes of storage are required for every IRC batch terminal.
ZCTCTTES	contains the small terminal entries. 368 bytes of storage are required for every MRO session and console.
ZCTPEXT	the TPE extension.
ZC2RPL	contains the duplicate RPLs for active tasks. Each active task associated with a VTAM terminal requires 304 bytes.
ZCTCTUA	contains the TCTTE user area. It can be located in of the following DSAs: CDSA, UDSA, ECDSA, or EUDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANY|BELOW and the system initialization parameter, TCTUAKEY=CICS|USER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the *CICS/ESA Resource Definition Guide* for more information.

CICS subpools in the ESDSA

Table 34. CICS subpools in the ESDSA	
Subpool name	**Description**
LDEPGM	contains extended (31) bit dynamically loaded application programs and programs defined EXECKEY(USER).
LDERES	contains extended (31) bit resident application programs.
SMSHRU31	is used for many control blocks of SHARED_USER24 class storage, RMI global work areas, EDF blocks for the life of the transaction being monitored, and other control blocks.
ZCTCTUA	contains the TCTTE user area. It can be located in of the following DSAs: CDSA, UDSA, ECDSA, or EUDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANY|BELOW and the system initialization parameter, TCTUAKEY=CICS|USER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the *CICS/ESA Resource Definition Guide* for more information.

CICS subpools in the ERDSA

Table 35. CICS subpools in the ERDSA

Subpool name	Description
LDENUCRO	contains the extended CICS nucleus and 31-bit macro tables. The extended CICS nucleus is approximately 1850KB. **APAR PQ07674** MJO 26/08/98 Contains programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited REENTRANT and RMODE(24).
LDENRSRO	contains the extended CICS nucleus and 31-bit macro tables. The extended CICS nucleus is approximately 1850KB. Contains programs defined EXECKEY(CICS) which are RESIDENT, that were link edited REENTRANT and RMODE(24).
LDEPGMRO	contains extended (31) bit dynamically loaded application programs. The contents of this subpool has to linked reentrant.
LDERESRO	contains extended (31) bit resident application programs. The contents of this subpool has to linked reentrant.

Short-on-storage conditions caused by subpool storage fragmentation

When migrating to CICS/ESA you may experience short-on-storage problems in the Dynamic Storage Areas (DSAs) below the 16M line. In many cases the amount of storage allocated to the region is greater than in previous releases.

CICS storage management incorporates support transaction isolation (subspaces) in CICS/ESA, including the fact that the DSAs are now managed by CICS with the DSA/EDSA limits being specified in the SIT, rather than a size for each DSA as in previous releases. See “The dynamic storage areas” on page 503 for a description of the DSAs and the subpools within each DSA.

Storage extents support dynamic storage management and provide subspace support. Storage extents are always allocated in multiples of 256K below the 16M line, with the exception of the UDSA which is allocated in 1M extents when transaction isolation is in use. Above the line extents are allocated in multiples of 1M.

When a DSA, such as the CDSA, requires additional storage in order to satisfy a GETMAIN request, the CICS storage manager allocates another extent to that DSA. However, if all extents are currently allocated, an attempt is made to locate a free extent in another DSA which may then be relocated to the DSA in need. However, in order to remove an extent from one DSA so that it may be allocated to another, all pages in the extent must be free (that is, not allocated to any subpool).

Analysis of short-on-storage problems begins by obtaining a dump when the system is in a short-on-storage condition. The best documentation is obtained by setting an entry in the dump table causing a dump to be taken when the DFHSM0131 (short-on-storage below the line) or DFHSM0133 (short-on-storage above the line) is issued. Use the CICS command CEMT SET SYDUMPCODE(SM0131) SYSDUMP MAXIMUM(1) ADD to indicate that a dump should be taken the first time a DFHSM0131 message is issued.

Use the IPCS command VERBX CICS520 'SM=3' to format the SM control blocks. Examine the DSA summaries, noting which DSA(s) are short-on-storage and the amount of free space in the other DSAs (above or below the 16M line as appropriate). The amount of freespace is given for each extent for each DSA,

Frequently either the UDSA or the CDSA is short-on-storage but there is a large amount of free storage in the SDSA. The following dump extracts are from a problem of this type where the UDSA is short-on-storage.

Each extent has an associated page pool extent (PPX) and page allocation map (PAM). Examination of the SDSA extents shows several extents with large amounts of freespace. For example, the extent beginning at 00700000 running through 0073FFFF has only 4K allocated and 252K free.

```
Extent list:      Start      End          Size     Free
                  00700000   0073FFFF     256K     252K
```

The DSA extent summary shows that the PPX for the extent at 00700000 is found at 09F0A100, and the associated PAM is found at 09F0A150. Examination of the PAM shows only one page is allocated, and it belongs to the subpool with an ID of x'7A'.

```
Start      End          Size   PPX_addr   Acc     DSA
00700000   0073FFFF     256K   09F0A100    C     SDSA

PPX.SDSA 09F0A100 Pagepool Extent Control Area

   0000  00506EC4 C6C8E2D4 D7D7E740 40404040  *.&>DFHSMPPX     *
   0010  E2C4E2C1 40404040 09A1BA68 071B3EA0  *SDSA    ........*
   0020  00040000 00700000 0073FFFF 071B5EE0  *.................*
   0030  00000000 09F0A150 00000040 0710A268  *.....0.&... ..s.*
   0040  0003F000 00000000 00000000 00000000  *..0.............*

PAM.SDSA 09F0A150 Page Allocation Map

   0000  00000000 00000000 00000000 00000000  *.................*
   0010 -    002F LINES SAME AS ABOVE
   0030  00000000 0000007A 00000000 00000000  *.................*
```

The domain subpool summary determines for the SDSA which subpool is associated with the ID of x'7A'. In this dump 7A is the ID for subpool ZCTCTUA. Do not rely on the IDs being the same for multiple runs of CICS because the IDs are assigned in the order the ADD_SUBPOOL is issued.

```
==SM: UDSA Summary (first part only)

   Size:                            512K
   Cushion size:                     64K
   Current free space:               56K  (10%)
 * Lwm free space:                   12K  ( 2%)
 * Hwm free space:                  276K  (53%)
   Largest free area:                56K
 * Times nostg returned:              0
 * Times request suspended:           0
   Current suspended:                 0
 * Hwm suspended:                     0
 * Times cushion released:            1
   Currently SOS:                   YES

==SM: SDSA Summary (first part only)

   Size:                           4352K
   Cushion size:                     64K
   Current free space:             2396K  (55%)
 * Lwm free space:                  760K  (17%)
 * Hwm free space:                 2396K  (55%)
   Largest free area:               252K
 * Times nostg returned:              0
 * Times request suspended:           0
   Current suspended:                 0
 * Hwm suspended:                     0
 * Times cushion released:            0
   Currently SOS:                    NO
```

A short-on-storage condition can occur as a result of large amounts of redundant program storage (RPS). This can be identified in the domain subpool summary and the loader domain summary (use the IPCS command VERBX CICS 410 'LD=3' to format the LD control blocks).

DFH0STAT provides useful information in the storage summary without a breakdown by subpool. DFH0STAT should be run just prior to the completion of

the statistics interval. For example, if the statistics interval is 3 hours, run DFH0STAT at 2 hours and 59 minutes. See “Storage Reports” on page 450 for more information.

APAR PQ07674 MJO 26/08/98

To ease the short-on-storage problems, you can add records to the local catalog to enable the CICS self-tuning mechanism for storage manager domain subpools. For details of how to do this using the CICS-supplied utility program, DFHSMUTL, see the see the *CICS/ESA Operations and Utilities Guide*. Alternatively, you can fix the size of a DSA using one or more of the following SIT overrides (see the *CICS/ESA System Definition Guide*):

CDSASZE
UDSASZE
SDSASZE
RDSASZE
ECDSASZE
EUDSASZE
ESDSASZE
ERDSASZE

The self- tuning mechanism and the SIT overrides should only be used if the short-on-storage problems are not completely resolved by increasing the DSA or EDSA limits.

Storage management requests the loader to reduce the RPS storage below 80%. This makes additional extents available to be allocated to the DSA in need.

LDPGMRO storage is allocated on a 16-byte boundary to reduce free space between programs.

If short-on-storage problems persist, initial DSA sizes may be specified as SIT overrides. The following process can be used to determine the values to use.

- Collect DFH0STAT output as described, for information showing storage use by DSA during the intervals.
- Review the CICS shutdown statistics for several days.

This provides information which can be used to define the amount of storage used at a subpool and a DSA level. Extent usage is shown with the number of extents added and released.

In addition to the DSA information provided in DFH0STAT, the shutdown information provides detailed information about each subpool, including the DSA where it was allocated. If statistics are being gathered, shutdown statistics will only provide data since the last statistics collection.

Determine if DSALIM has been specified as large as possible, but allowing for OSCORE requirements of the various packages in use.

Allocating into managed extents can lead to a block of storage in an extent which is insufficient to satisfy a getmain request. With the dynamic nature of the subpools and DSAs, this should be relieved as the subpool/extent storage is reused. Specifying the initial DSA size using the SIT override for the affected DSA has the

effect of reserving contiguous extents up to the amount specified, and eliminating the blocks of storage.

Additional DSAs (RDSA and SDSA) are available and many of the subpools from the UDSA are moved to the SDSA. The shutdown statistics or information in a dump of the CICS region can be used to define relative sizes of the subpools and associated DSAs.

Also, using the LPA reduces the amount of storage used in LDNUCRO by approximately 100K.

MAPS should be defined as MAPS. Defining MAPS as programs causes them to be loaded into LDRES rather than in LDNUC. LDRES is part of the SDSA and more sensitive to fragmentation. For PSBPOOL space, the shutdown statistics provide the correct size.

CICS kernel storage

CICS kernel storage consists of control blocks and data areas that CICS requires to manage system and user tasks throughout CICS execution. The majority of this storage is allocated from the CICS DSAs. A small amount of this storage is allocated from MVS storage.

The kernel recognises two types of task: static tasks, and dynamic tasks. The kernel storage for static tasks is pre-allocated and is used for tasks controlled by the MXT mechanism. The storage for dynamic tasks is not pre-allocated and is used for tasks such as system tasks which are not controlled by the MXT value. Because the storage for dynamic tasks is not pre-allocated, the kernel may need to GETMAIN the storage required to attach a dynamic task when the task is attached.

The number of static tasks is dependant upon the current MXT value (there are MXT+1 static tasks). The storage for static tasks is always GETMAINed from the CICS DSAs. If MXT is lowered the storage for an excess number of static tasks is freed again.

During early CICS initialization the kernel allocates storage for 8 dynamic tasks. This storage is GETMAINed from MVS and is always available for use by internal CICS tasks. All other storage for dynamic tasks is then allocated, as needed, from the CICS DSAs. Typically when a dynamic task ends, its associated storage is freed.

The storage required by a single task is the same for both types of task and can be divided into storage required above and below the 16MB line:

- Above the line the following storage is required per task:
  - A 896-byte kernel task entry
  - A 24K 31-bit stack.
- Below the line the following storage is required per task:
  - A 2K 24-bit stack.

In addition to this storage, the kernel also allocates a number of 4K extension stacks both above and below the 16MB line. These are for use by any task, if it overflows the stack storage allocated to it. The number of 24-bit and 31-bit stack

extensions pre-allocated by the kernel is determined by dividing the current MXT value by 10.

When the kernel GETMAINs storage from the CICS DSAs, the following subpools are used:

- In the CDSA:

```
KESTK24             2K stack segments
KESTK24E            4K extension stack segments
```

- In the ECDSA:

```
KESTK31             24K stack segments
KESTK31E            4K extension stack segments
KETASK              896 byte task entries
```

End of Diagnosis, Modification or Tuning Information

Appendix D. Performance data

This appendix contains:

- Timings for various CICS functions
- Timings for MRO and ISC
- Sizes of CICS tables located below the 16MB line.

Estimated processor timings for CICS functions

This section contains the following tables:

These tables allow you to make comparisons between the various CICS functions and some of the factors that affect the processing time of particular CICS functions. These tables can also help you to make decisions concerning application design when you are considering performance. To calculate a time for a function, find the entries appropriate to your installation and application, and add their values together.

Notes on the tables

1. All the processing times are estimates based on measurements of similar functions, and are in **milliseconds** for a hypothetical processor that would be equivalent in power to a System/370 Model 4341-11. These timings are for execution of CICS functions within the CICS region.
2. The processing times will increase when you use CICS monitoring or trace facilities. The increases due to the CICS monitoring facility vary with the transaction mix, system definition, and system configuration. Measurements of CICS workloads under laboratory conditions have shown that the overhead of performance class monitoring is in the range of 7 to 11%.
3. Similarly, for internal trace, the processing increase varies on different transaction mixes. Trace selectivity allows a wide range of tracing activity to be specified. The associated overhead will clearly also have a wide range. Estimates included here are for level 1 tracing for all CICS components which is the functional equivalent of CICS/MVS Version 2.1 internal trace.
4. All these timings assume no paging. However, if a reference is made to a virtual storage page that does not reside in real storage, operating-system time will be used to retrieve the required page.
5. NE. No estimate is available.
6. N/A. Not applicable.
7. Recoverability assumes journal on DASD using files with RECOV set to BACKOUTONLY (FCT with LOG=YES).

Transaction initialization

Table 36. Transaction initialization

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
	Via VTAM
Read in and attach task search for transaction	13.28
Initialize	
Assembler	1.21
COBOL II	4.60
Internal Trace	6.87

Transaction termination

Table 37. Transaction termination

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
	Via VTAM
Return EXEC/CICS clean-up Assembler COBOL II	 1.54 4.00
LUW syncpoint	9.74
Internal Trace	4.49
Terminate transaction Deferred transaction (per request unit(RU))	 9.25

Note: These timings are for when NOWAIT parameter is used on any previous EXEC SEND commands (see "Send to 3270" on page 523).

Send to 3270

Table 38. Send to 3270

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
	Via VTAM
EXEC SENDMAP map fields Assumes NOWAIT	2.90 See note 1. See note 2.
Internal trace	2.44.

Notes:

1. The time taken to map the fields depends on the number of fields and the number of maps.

   The map fields timing is:

   ```
   (0.70 x nm) + (0.08 x nf)
   ```

   where

   - "nm" is the number of floating maps.
   - "nf" is the number of fields in the map. This number should include application data fields, constant fields, and null fields (that is, the spaces between visible fields).

2. See Table 37 on page 523 for timings for NOWAIT parameter

RECEIVE from 3270

Table 39. RECEIVE from 3270

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
	Via VTAM
EXEC RECEIVEMAP locate map unmap fields	1.48 0.79 See note 1.
Internal trace	2.78
Flush pending sends Terminal Control (per RU). See note 2.	10.80 11.42

Notes:

Performance timings are for receive map with one input field against a map with 80 fields (3 RUs).

1. The time to unmap the fields depends upon the number of fields and the number of maps. The timing to unmap the screen input fields is:

   `(0.56 x nm) + (rf x 0.02mf) + (0.08rf)`

   where:

   - "nm" is the number of maps.
   - "mf" is the number of fields to be checked (left to right, top to bottom) before the correct field position in the map is found; repeated for each received field.
   - "rf" is the number of received fields.

2. These timings vary depending on the length of the transmitted message. Terminal control timing is:

   `INTEGER(message/RU) (7.62)`

WRITEQ transient data (VSAM)

Table 40. WRITEQ transient data (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
	Intrapartition
WRITEQ TD (see notes 1 and 2) With I/O Buffering only	 10.6 4.4
Internal trace	1.3
Recoverability no trace With internal trace	 7.49 8.90

Notes:

1. **Single Buffer Support**: These times vary depending on the sequence of writes to a particular queue or to a different queue. If the writes are to the same queue, the time is the sum of one WRITEQ TD with I/O plus (n–1) without I/O, where *n* is the number of records that can fit the transient data CI size.

   If the WRITEQ TD is to a different queue (worst case), the time is the sum of one WRITEQ TD to flush out the previous queue buffer, plus a READQ TD if the QUEUE already exists, plus a further WRITEQ TD without I/O.

2. **Multiple Buffer Support**: The aim of multiple buffer support is to:

   - Reduce the amount of I/O to service requests, excluding recoverability
   - Allow a degree of concurrency of I/O operations.

   If you specify only a single string and buffer, execution will be as described in Note 1. If you specify more than one buffer, this increases the probability of a buffer being available for the WRITEQ TD data and, therefore, minimizes the output operation. If recovery is specified, the output operations always occur.

   If all buffers are full, output operations take place to free up buffers for the current request.

READQ transient data (VSAM)

Table 41. READQ transient data (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
	Intrapartition
READQ TD	
With I/O	6.53
From buffer only (see note 1)	1.80
Internal trace	1.3
Recoverability	
no trace	7.49
With internal trace	8.90

Notes:

The aim of multiple buffer support is to:

1. Reduce the amount of I/O to service requests, excluding recoverability
2. Allow a degree of concurrency of I/O operations.

If you specify more than one buffer, this increases the probability of the required data existing in one of the buffers.

If the data cannot be found in the buffers, one or more I/O operations are necessary. If the data buffers are not “mirrored” on the output device, a buffer is written to free the buffer for the read operation. The read operation is then performed.

WRITEQ temporary storage

Table 42. WRITEQ temporary storage

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)	
	Main Storage	**Aux. Storage**
WRITEQ TS		
With I/O	N/A	7.19
Without I/O	2.29	2.56
Recoverability	N/A	7.49
no trace	N/A	8.90
With internal trace		
Internal trace	1.03	1.03

READQ temporary storage

Table 43. READQ temporary storage

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)	
	Main Storage	Aux. Storage
READQ TS		
With I/O	N/A	8.59
Without I/O	1.78	1.99
Recoverability		
no trace	N/A	7.49
With internal trace	N/A	8.90
Internal trace	1.03	1.03

START BROWSE (VSAM)

Table 44. START BROWSE (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
START BROWSE	2.89
Index I/O	7.40
Sequence set I/O	7.40
Data buffer I/O	7.40
Internal trace	0.73

READ NEXT BROWSE (VSAM)

Table 45. READ NEXT BROWSE (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
READ NEXT	
Read with I/O	6.63
From buffer only	1.42
Internal trace	0.37

END BROWSE (VSAM)

Table 46. END BROWSE (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
END BROWSE	1.26
Internal trace	0.37

READ (VSAM)

Table 47. READ (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)	
	K S D S	E S D S
READ		
Index with I/O	7.40	N/A
Sequence set I/O	7.40	N/A
Read data with I/O	9.86	9.31
From Buffer only	3.09	2.21
Recovery (see note)	1.03	NE
Internal trace from buffers only	0.38	0.38

Note: When RECOV is set to BACKOUTONLY (AUTOLOG=YES in FCT), the before image is placed in the journal buffer by the READ for UPDATE. It is not forced out to DASD until the REWRITE, by which time it may have been forced out by other journal activity.

WRITE (VSAM)

Table 48. WRITE (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)	
	K S D S	E S D S
WRITE		
Index with I/O	NE	
Sequence set I/O	NE	
Data with I/O	11.74	
Recoverability		
no trace	11.42	
with internal trace	13.14	
Internal trace	0.59	

Note: CICS file integrity includes a READ operation prior to the WRITE to check whether the record exists.

REWRITE (VSAM)

Table 49. REWRITE (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)	
	K S D S	E S D S
WRITE		
Index with I/O	7.83	N/A
Sequence set I/O	7.83	N/A
Data with I/O	9.24	9.18
Data only	NE	NE
Recoverability		
no trace	7.49	7.49
With internal trace	8.90	8.90
Internal trace	0.58	0.58

DELETE (VSAM)

Table 50. DELETE (VSAM)

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
	KSDS
DELETE	11.46
Recoverability	
no trace	7.49
With internal trace	8.90
Internal trace	0.59

Storage control

Table 51. Storage control

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
GETMAIN	0.80
FREEMAIN	0.78
Internal trace	
GETMAIN	0.41
FREEMAIN	0.46

Program control

Table 52. Program control

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
LINK	
Assembler	1.41
COBOL II	4.07
Internal trace	1.60
RETURN	
Assembler	1.11
COBOL II	3.44
Internal trace	0.87

User journaling

Table 53. User journaling

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
EXEC interface (STARTIO=YES)	
Write to log	4.59
Internal trace	1.12

ENQ/DEQ resource

Table 54. ENQ/DEQ resource

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
ENQ	0.23
DEQ	0.22
Internal trace	0.59

Interval control

Table 55. Interval control

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
SUSPEND	0.66
ASKTIME	0.12
Internal trace	
SUSPEND	0.64
ASKTIME	0.41

EXEC conditions

Table 56. EXEC conditions

CICS INTERNAL FUNCTION	APPROXIMATE TIMINGS (ms)
EXEC functions	
HANDLE AID	0.43
HANDLE CONDITION	0.38
IGNORE	0.32
ASSIGN ABCODE	0.16
ADDRESS CWA	0.16
Internal trace	
Internal storage (aid)	0.38
(cond)	0.40
(ign)	0.40
(abc)	0.20
(cwa)	0.20

MRO

Estimated processor timings for function shipping on MRO

Table 57. Estimated timings for function shipping on MRO

CICS INTERNAL FUNCTION	Application-owning region	Resource-owing region
First function ship	7.7	12.0
Subsequent function ship	5.20	5.75
Last function ship including syncpoint	15.50	14.50
Low utilization effect	1.40	1.40

Notes:

1. The timings were obtained from an investigation of shipped file requests. Other types of requests may show slightly different characteristics.
2. These times should be added to the processing times of the CICS commands that are function shipped.
3. The low utilization effect is a consequence of the extra work involved in task communication and management when the system is not busy. The timings given here are an estimate of this effect. They should be added for the percentage of function shipped flows which arrive at either region and find it in an MVS WAIT.
4. The syncpoint flow is for a two region MRO case where the file owning region (FOR) has a logical unit of work to commit.

Glossary

This glossary defines CICS terms used in this book and words used with other than their everyday meaning. In some cases, a definition may not be the only one applicable to a term, but gives the particular sense in which it is used in this book.

This glossary includes terms and definitions from the *IBM Vocabulary for Data Processing, Telecommunications, and Office Systems*, GC20-1699.

American National Standards Institute (ANSI) definitions are preceded by an asterisk (*).

The symbol "(ISO)" at the beginning of a definition indicates that it has been discussed and agreed on at meetings of the International Organization for Standardization, Technical Committee 97/Subcommittee 1, and has been approved by ANSI for inclusion in the *American National Dictionary for Information Processing*.

A

abend. Abnormal end of task.

ACB. Access method control block (VTAM and VSAM).

access key. An indicator associated with a reference to storage. It is matched to the **storage key** to permit access to the storage. In most cases, the access key is the PSW key in the current PSW.

access method. A technique for moving data between main storage and input/output devices.

access method control block (ACB). A control block that links an application program (for example, a CICS program) to an access method (for example VSAM or ACF/VTAM). An ACB is used when communicating with DL/I only when the underlying access method is VSAM.

ACF. Advanced Communications Function.

active session. In XRF, a session between a class 1 terminal and the active system. A session that connects the active CICS to an end user.

active system. In an XRF environment, the CICS system that currently supports the processing requests of the user.

active task. (1) A CICS task that is eligible for dispatching by CICS. (2) During emergency restart, a task that completed an LUW and started another, but that did not cause any records to be written to the system log during the second LUW. During recovery-control processing, an LUW completion but no physical end-of-task (that is, task DETACH) is found.

activity keypoint. A record of task and DCT entry status on the system log made on a periodic basis to facilitate the identification of transaction backout information during emergency restart. In the event of an uncontrolled shutdown and subsequent emergency restart, activity keypoints can shorten the process of backward scanning through the system log. Activity keypoints are written automatically by the system (**system activity keypoints**) or by the user (**user activity keypoints**).

address space. The complete range of addresses that is available to a program.

addressing mode (AMODE). In MVS, the mode, 24-bit or 31-bit, in which a program stores addresses. The AMODE linkage-editor control statement specifies the addressing mode of the load module produced.

Advanced Communications Function (ACF). A group of program products for users of MVS that can improve single-domain and, optionally, multidomain data communication capability.

Advanced Program-to-Program Communication (APPC). The SNA protocol boundary of the presentation services layer of the LU6.2 architecture. APPC is commonly used as a synonym for LU6.2.

after image. A record of the contents of a data element after it has been changed. After images are used for forward recovery.

agent. In a two-phase commit or MRO syncpointing sequence, a task that receives syncpoint requests from the initiator (the task that initiates the syncpoint activity).

AID. Automatic Initiate Descriptor.

AIEXIT. System initialization parameter used to specify the name of the autoinstall user program that you want CICS to use when autoinstalling VTAM terminals. The default is the name of the CICS-supplied autoinstall user program, DFHZATDX. See the *CICS/ESA System Definition Guide* for more information.

AILDELAY. System initialization parameter used to specify the delay period that elapses between the end of a session between CICS and a terminal and the deletion of the terminal entry. The default is zero, meaning that the terminal entry is deleted as soon as

the session is ended. See the *CICS/ESA System Definition Guide* for more information.

AIQMAX. System initialization parameter used to specify the maximum number of devices that can be queued concurrently for autoinstall. The default is 100. See the *CICS/ESA System Definition Guide* for more information.

AIRDELAY. System initialization parameter used to specify the delay period that elapses after an emergency restart, before autoinstalled terminals that are not in session are deleted. The default is 700, meaning a delay of seven minutes. See the *CICS/ESA System Definition Guide* for more information.

AKPFREQ. System initialization parameter used to specify the frequency of activity keypoints. The default is 200. See the *CICS/ESA System Definition Guide* for more information.

alternate system. In an XRF environment, alternate describes the system that is standing by to take over the workload when the active CICS system fails or a takeover is initiated.

alternate system. In an XRF environment, a CICS system that stands by to take over the user workload when the active CICS system fails or a takeover is initiated.

AMODE. In MVS, a control statement that defines the addressing mode of the load module produced by the linkage editor. See **addressing mode**.

APAR. Authorized program analysis report.

APPC. Advanced Program-to-Program Communication.

APPLID. System initialization parameter used to specify the VTAM application identifiers (applids) for this CICS region. See the *CICS/ESA System Definition Guide* for more information.

assembler language. A source language that includes symbolic machine-language statements in which there is a one-to-one correspondence with the instruction formats and the data formats of the computer. Before execution, a CICS assembler-language application program must be processed by the translator, assembler, and linkage editor. See also **FORMATTIME**.

asynchronous processing. A means of distributing the processing of an application between systems in an intercommunication environment. The processing in each system is independent of the session on which requests are sent and replies are received. No direct correlation can be made between requests and replies and no assumptions can be made about the timing of the replies.

automatic initiate descriptor (AID). A control block used internally by CICS for scheduling purposes. An example of AID use is scheduling a transaction, optionally associating it with a terminal and a temporary storage queue. Another use is scheduling MRO, LU6.1, and LU6.2 ALLOCATE requests.

automatic transaction initiation (ATI). The initiation of a CICS transaction by an internally-generated request, for example, the issue of an EXEC CICS START command or the reaching of a transient data trigger level. A transaction can be initiated immediately, at a specified time, after a specified time interval, or, if a terminal is required, as soon as that terminal is free.

auxiliary storage. Data storage other than main storage; for example, storage on magnetic tape or direct access devices.

auxiliary trace. An optional CICS function that causes trace entries to be recorded in the auxiliary trace data set, a sequential data set on disk or tape.

AUXTR. System initialization parameter used to indicate whether the auxiliary trace destination is to be activated at system initialization. This parameter controls whether any of the three types of CICS trace entry (system trace, user trace, and exception trace) are written to the auxiliary trace data set. See the *CICS/ESA System Definition Guide* for more information.

availability. The degree to which a system or resource is ready when needed to process data; the *percentage* of time a system, network, or component can be utilized, within a certain time frame. Generally, the percentage is derived by dividing **actual availability** time by **scheduled availability** time.

average throughput rate. The power of a system to process a representative work load. The power of the system is measured in units of data processing work, for example, jobs or transactions successfully completed per hour, minute, or second.

AVM. Availability manager.

B

backout. Restoring to a previous state all or part of a system. The process of removing all the updates against **protected resources** such as files and DL/I databases performed by an application program that either has terminated abnormally or was in-flight at the time of a CICS or MVS image failure. Backout can by done dynamically in the case of an application abend,

or during restart in the case of CICS or MVS image failure.

backout. The process of restoring to a previous state all or part of a system. The process of removing all the updates against **protected resources** such as files and DL/I databases performed by an application program that either has terminated abnormally or was inflight at the time of a CICS or MVS image failure. Backout can by done dynamically in the case of an application abend, or during restart in the case of CICS or MVS failure.

backup session. The session built by VTAM to the alternate CICS system for XRF-capable terminals, used after a takeover to reestablish service to the terminals.

backup session. In XRF, the session built by VTAM to the alternate CICS system for XRF-capable terminals, used after a takeover to reestablish service to the terminals.

Basic Direct Access Method (BDAM). An access method used to retrieve or update particular blocks of a data set on a direct access device.

Basic direct access method (BDAM). An access method used to retrieve or update particular blocks of a data set on a direct access device.

basic mapping support (BMS). A facility that moves data streams to and from a terminal. BMS is an interface between CICS and its application programs. It formats input and output display data in response to BMS commands in programs. To do this, it uses device information from CICS system tables and formatting information from maps you have prepared for your application programs.

BMS provides message routing, terminal paging, and device independent services. Most of the BMS programs are resident in the CICS nucleus.

BMS exists in three pregenerated versions: minimum, standard, and full function. Each version provides a different level of function, and therefore

Basic sequential access method (BSAM). An access method for storing or retrieving data blocks in a continuous sequence.

Basic telecommunications access method (BTAM). An access method that enables read/write communication with remote devices.

BDAM. Basic direct access method.

Binary synchronous communication (BSC). Data transmission in which synchronization of characters is controlled by timing signals generated at the sending and receiving stations.

BMS. Basic Mapping Support.

BMS. System initialization parameter used to specify which version of basic mapping support (BMS) is to be included. The default is full-function BMS.

BookManager. A family of IBM products that enable users to create and display online books.

boundary network node (BNN). (1) In SNA, a subarea node that provides protocol support for adjacent peripheral nodes, for example, transforming network addresses to local addresses and *vice versa*, and providing session-level support for these peripheral nodes. (2) In XRF, the point at which terminal sessions are switched from the failing active system to the new active system. The communication controller at the BNN must be able to operate in an XRF configuration.

boundary network node (BNN). A subarea node (such as a 372X) that provides protocol support for adjacent peripheral nodes such as transforming network addresses to local addresses and vice versa, and providing session level support for these peripheral nodes.

BSAM. Basic sequential access method.

BSC. Binary synchronous communication.

C

C/370. A programming language designed for a wide range of system and commercial applications.

capacity planning. An analysis of processor loading and processor capacity, extending into real storage, other resources (channels, DASD, lines), and timings and response when necessary.

CAVM. CICS availability manager.

CDSA. CICS dynamic storage area.

CDSASZE. System initialization parameter used to specify the amount of storage to be allocated by CICS for the CICS dynamic storage area (CDSA) below the 16MB line. The default size is 1536KB. See the *CICS/ESA System Definition Guide* for more information.

CEBR. A CICS-supplied transaction used to browse temporary storage queues. See the *CICS/ESA CICS-Supplied Transactions* for more information.

CEBT. A CICS-supplied transaction that the operator can issue from the MVS console to control an alternate CICS system. See the *CICS/ESA CICS-Supplied Transactions* for more information.

CEDA. The main CICS-supplied transaction used to define resources online. Using CEDA, you can update both the CICS system definition data set (CSD) and the running CICS system. See the *CICS/ESA Resource Definition Guide* for more information.

CEMT. A CICS-supplied transaction used to invoke all the master terminal functions. These functions include inquiring and changing the value of parameters used by CICS, altering the status of system resources, terminating tasks, and shutting down CICS. See the *CICS/ESA CICS-Supplied Transactions* for more information.

CEST. A CICS-supplied transaction used to invoke a subset of the master terminal (CEMT) functions. CEST allows you to inquire about and alter some of the values of lines, netnames, tasks, and terminals. See the *CICS/ESA CICS-Supplied Transactions* for more information.

CETR. A CICS-supplied transaction used to control CICS tracing activity. See the *CICS/ESA CICS-Supplied Transactions* for more information.

checkpoint. In IMS, point at which an application program commits that the changes it has made to a database are consistent and complete, and releases database segments for use by other programs. You can request checkpoints at appropriate points in a program to provide places from which you can restart that program if it, or the system, fails.

For an IMS system, a point in time from which the system can start again if a failure makes recovery necessary. The checkpoint is performed by IMS itself.

CI. Control interval.

CICS availability manager (CAVM). In XRF, the mechanism that provides integrity for a CICS system with XRF. The CAVM uses the control file and the message file to handle communication between the active and alternate systems.

CICS dynamic storage area (CDSA). A storage area allocated below the 16MB line, intended primarily for the small amount of CICS code and control blocks that remain below the line in CICS/ESA 3.3. The size of the CDSA is controlled by the CDSASZE system initialization parameter.

CICS monitoring facility. The CICS monitoring facility (part of the system monitoring component) gives a comprehensive set of operational data for CICS, using one data recording program and, optionally, one or more data sets. See also **performance class data**, **exception class data**, and **SYSEVENT data**.

CICS monitoring facility data set. CICS monitoring facility data sets are used to record information that is output by the CICS monitoring facility program. The MCT defines which journal data sets are used by each class of monitoring. These data sets appear in the JCT with the keyword FORMAT=SMF. The format of the records is the system management facility (SMF), type 110 format.

CICS PD/MVS. CICS Problem Determination/MVS (program number 5695-035) is a set of online tools to help system programmers analyze and manage system dumps. It automates dump analysis and formats the results into interactive online panels that can be used for further diagnosis and resolution of problems.

CICS private area. Element of CICS storage that has both static and dynamic storage requirements. The static areas are set at initialization time and do not vary over the execution of that address space. The dynamic areas increase or decrease their allocations as the needs of the address space vary.

CICS program library. The CICS program library contains all user-written programs and CICS programs to be loaded and executed as part of the online system. This group includes the control system itself and certain user-defined system control tables essential to CICS operation. The library contains program text and, where applicable, a relocation dictionary for a program. The contents of this library are loaded asynchronously into CICS dynamic storage for online execution.

CICS region userid. The userid assigned to a CICS region at CICS initialization. It is specified **either** in the RACF started procedures table when CICS is started as a started task, **or** on the USER parameter of the JOB statement when CICS is started as a job.

CICS system definition data set (CSD). A VSAM KSDS cluster with alternate paths. The CSD data set contains a resource definition record for every record defined to CICS using resource definition online.

CICS-attachment facility. Provides a multithread connection to DB2 to allow applications running under CICS to execute DB2 commands.

CICS-key. Storage in the key in which CICS is given control (key 8) when CICS storage protection is used. It is for CICS code and control blocks and can be accessed and modified by CICS. Application programs in user-key cannot modify CICS-key storage, but they can read it. The storage is obtained in MVS exclusive-key storage. Compare with **user-key**..

CICSPARS/MVS. The Customer Information Control System Performance Analysis Reporting System (CICSPARS/MVS) (program number 5665-355) provides a method of reporting performance and accounting information produced by the CICS monitoring facility.

class 1 terminal. In XRF, a remote SNA VTAM terminal connected through a boundary network node IBM 3745/3725/3720 Communication Controller with an NCP that supports XRF. Such a terminal has a backup session to the alternate CICS system.

class 2 terminal. In XRF, a terminal belonging to a class mainly comprised of VTAM terminals that are not eligible for class 1. For these terminals, the alternate system tracks the session, and attempts reestablishment after takeover.

class 3 terminal. In XRF, a terminal belonging to a class mainly comprised of TCAM(DCB) terminals. These terminals lose their sessions at takeover.

CLT. System initialization parameter used to specify the suffix for the command list table, if this system initialization table is used by an alternate XRF system. See the *CICS/ESA System Definition Guide* for more information.

cluster. A data set defined to VSAM. A cluster can be a key-sequenced data set, an entry-sequenced data set, or a relative record data set.

CMAC. A CICS-supplied transaction used to display individual message information as it is provided in the CICS *CICS/ESA Messages and Codes* manual. See the *CICS/ESA CICS-Supplied Transactions* for more information.

CMXT. System initialization parameter used to specify the maximum number of tasks that can exist in any of the ten transaction classes in which you can categorize transactions. See the *CICS/ESA System Definition Guide* for more information.

COBOL. Common business-oriented language. An English-like programming language designed for business data processing applications.

command list table (CLT). In XRF, a CICS table that contains a list of MVS commands and messages to be issued during takeover. The CLT is defined to the alternate CICS system and used during takeover.

command security. A form of security checking that can be specified for a subset of the CICS application programming interface (API) commands. Command security operates in addition to any transaction security or resource security specified for a transaction. For example if a terminal invokes a transaction that the user is authorized to use, and the transaction issues a command that the user is not authorized to use, the command fails with the NOTAUTH condition.

COMMAREA. Communication area.

common system area (CSA). A major CICS storage control block that contains areas and data required for the operation of CICS. It can be extended to include a user-defined common work area (CWA) that can be referred to by application programs. This area is duplicated above the 16MB line as the **extended common system area (ECSA)**.

common work area (CWA). The common work area (CWA) is an area within the CSA that can be used by application programs for user data that needs to be accessed by any task in the system. This area is acquired during system initialization and its size is determined by the system programmer at system generation. It is initially set to binary zeros. Its contents can be accessed and altered by any task during CICS operation. Contrast with **transaction work area (TWA)**.

communication area (COMMAREA). An area that is used to pass data between tasks that communicate with a given terminal. The area can also be used to pass data between programs within a task.

communication management configuration (CMC). A configuration in which the VTAM subsystem that owns the terminals is in a different MVS image from the active or the alternate CICS system.

constraint. This is sometimes referred to as "transaction throughput degradation" or "bottleneck" – a place in the system where contention for a resource is affecting performance.

control block. In CICS, a storage area used to hold dynamic data during the execution of control programs and application programs. Synonym for **control area**. Contrast with **control table**.

control data set. A data set that ensures XRF system integrity by allowing only one active CICS system to access a particular set of resources. It is used by the active and the alternate CICS systems to monitor each other's well-being.

control interval (CI). (1) A fixed-length area of auxiliary-storage space in which VSAM stores records and distributes free space. (2) The unit of information transmitted to or from auxiliary storage by VSAM, independent of physical record size.

control subpool. A CICS area that holds the dispatch control area (DCA), interval control elements (ICEs), automatic initiate descriptors (AIDs), queue element areas (QEAs), and other control information. Generally, the control subpool occupies only one page.

control table. In CICS, a storage area used to define or describe the configuration or operation of the system. Contrast with **control block**.

control terminal. In CICS, the terminal at which a designated control operator is signed-on.

conversational. Pertaining to a program or a system that carries on a dialog with a terminal user, alternately accepting input and responding to the input quickly enough for the user to maintain a train of thought.

CSA. Common system area.

CSAC. Transient data destination used by the abnormal condition program (DFHACP).

CSD. CICS system definition data set.

CSMT. Transient data destination used by the terminal abnormal condition program (DFHTACP), the node abnormal condition program (DFHZNAC), and the abnormal condition program (DFHACP) for writing terminal error and abend messages.

CSTE. Transient data destination used by the terminal abnormal condition program (DFHTACP).

CWA. Common work area.

CWAKEY. System initialization parameter used to specify the storage key for the CWA if CICS is running with the storage protection facility. The default storage key is user-key.

D

DASD. Direct access storage device.

data availability. An IMS enhancement available with DBCTL. It allows PSB scheduling to complete successfully even if some of the full-function databases it requires are not available.

Data control block (DCB). An MVS control block used by access method routines in storing and retrieving data.

data element. The smallest unit of data that can be referred to. Synonymous with **field**.

data entry database (DEDB). An IMS hierarchic database designed to provide efficient storage and fast online gathering, retrieval, and update of data using VSAM ESDS. From CICS/ESA, a DEDB is accessible only through DBCTL, not through local DL/I.

Data Language/I (DL/I). A high-level interface between applications and IMS. It is invoked from PL/I, COBOL, or assembler language by means of ordinary subroutine calls. DL/I enables you to define data structures, to relate structures to the application, and to load and reorganize these structures. It enables applications programs to retrieve, replace, delete, and add segments to databases.

data management block (DMB). An IMS control block that resides in main storage and describes and controls a physical database. It is constructed from information obtained from the application control block (ACB) library or the database description (DBD) library.

data set name sharing. An MVS option that allows one set of control blocks to be used for the base and the path in a VSAM alternate index.

data sharing (IMS). The concurrent access of DL/I databases by two or more IMS/VS subsystems. The subsystems can be in one processor or in separate processors. In IMS data sharing, CICS/ESA can be an IMS subsystem. There are two levels of data sharing: **block-level data sharing** and **database-level data sharing**.

data stream. All information (data and control information) transmitted through a data channel in a single read or write operation.

data-owning region (DOR). A CICS address space whose primary purpose is to manage files and databases. See **application-owning region (AOR)**, and **terminal-owning region (TOR)**.

DATABASE 2 (DB2). A relational database management system in which data is presented to the user in the form of tables.

database-level sharing. A kind of IMS data sharing that enables application programs in one IMS system to read data while a program in another IMS system reads it or updates it.

DBUFSZ. System initialization parameter coded with a value that CICS can use in an internal algorithm that calculates the size of the dynamic log buffer needed for dynamic transaction backout. The default is 500.

DCT. System initialization parameter used to specify the destination control table suffix. See the *CICS/ESA System Definition Guide* for more information.

deadlock. (1) Unresolved contention for the use of a resource. (2) An error condition in which processing cannot continue because each of two elements of the process is waiting for an action by, or a response from, the other.

deferred work element (DWE). A work element created and placed on a chain (the DWE chain) to save information about an event that must be completed before task termination but is not completed at the present time. DWEs are also used to save information about work to be backed out in case of an abend.

destination control table (DCT). A table containing an entry for each extrapartition, intrapartition, and

indirect transient data destination used in the system, or in connected CICS systems.

device independence. The capability to write application programs so that they do not depend on the physical characteristics of devices. BMS provides a measure of device independence.

DFH. Three-character prefix of all CICS modules.

DFHCSDUP. CICS system definition data set (CSD) utility program. It provides offline services for the CSD. It can be invoked as a batch program or from a user-written program running either in batch mode or under TSO.

dispatching. The act of scheduling a task for execution, performed by CICS task control.

dispatching priority. A number assigned to tasks, used to determine the order in which they are to use the processor in the CICS multitasking environment.

distributed program link (DPL). Type of CICS intercommunication which, in CICS/ESA 3.3, enables CICS/ESA to ship LINK requests between host CICS regions. In CICS OS/2, DPL enables CICS OS/2 to ship LINK requests up to a host CICS region, or to another CICS OS/2 system.

distributed transaction processing (DTP). Type of intercommunication in CICS, in which the processing is distributed between transactions that communicate synchronously with one another over intersystem or interregion links.

DLI. System initialization parameter used to indicate whether DL/I databases are to be accessed, using CICS local or remote DL/I support, during the execution of CICS. See the *CICS/ESA System Definition Guide* for more information.

DLMON. System initialization parameter used to specify that DL/I database monitoring using the IMS database monitor is to be active for this invocation of CICS. The default is NO. This parameter is only applicable if you have also coded DLI=YES. See the *CICS/ESA System Definition Guide* for more information.

DLTHRED. System initialization parameter used to specify the number of threads provided through the CICS local DL/I interface. The default is 1. See the *CICS/ESA System Definition Guide* for more information.

DMB. Data management block (DL/I).

DMBPL. System initialization parameter used to specify the data management block pool size in 1024-byte blocks for CICS-DL/I interface support. The default number of blocks is four. See the *CICS/ESA System Definition Guide* for more information.

DPL. Distributed program link.

DTB. Dynamic transaction backout.

DTP. Distributed transaction processing.

DUMP. System initialization parameter used to specify whether CICS is to take SDUMPs. The default is YES. See the *CICS/ESA System Definition Guide* for more information.

dump control. The CICS element that provides storage dumps for help during testing.

dynamic transaction backout (DTB). The process of canceling changes made by a transaction to stored data following the failure of that transaction for whatever reason. Dynamic transaction backout is required with resource definition online.

E

ECDSA. Extended CICS dynamic storage area.

ECDSASZE. System initialization parameter used to specify the amount of storage to be allocated by CICS for the extended CICS dynamic storage area (ECDSA) above the 16MB line. The default is 8MB. See the *CICS/ESA System Definition Guide* for more information.

ECSCS. System initialization parameter used to specify how much of the extended CICS dynamic storage area (ECDSA) is to be reserved for the storage cushion. See the *CICS/ESA System Definition Guide* for more information.

emergency restart. The CICS backout facility for an automatic restart following a system failure. It restores the recoverable resources of all interrupted transactions to the condition they were in when they started.

end-of-day statistics. CICS statistics collected since the last event that involved a reset, such as a shutdown.

enqueued. The state of a task scheduled to update a physical segment of a database when another task is currently accessing that segment.

EPDM. Enterprise Performance Data Manager/MVS

ERDSASZE. System initialization parameter used to specify the amount of storage to be allocated by CICS for the extended read-only dynamic storage area (ERDSA) above the 16MB line. The default is 3MB.

See the *CICS/ESA System Definition Guide* for more information.

ERSCS. System initialization parameter used to specify how much of the extended read-only dynamic storage area (ERDSA) is to be reserved for the storage cushion. See the *CICS/ESA System Definition Guide* for more information.

EUDSASZE. System initialization parameter used to specify the amount of storage to be allocated by CICS for the extended user dynamic storage area (EUDSA) above the 16MB line. The default is 8MB. See the *CICS/ESA System Definition Guide* for more information.

EUSCS. System initialization parameter used to specify how much of the extended user dynamic storage area (EUDSA) is to be reserved for the storage cushion. See the *CICS/ESA System Definition Guide* for more information.

Event control block (ECB). An MVS control block that represents the status of an event. CICS task control uses ECBs.

event monitoring point (EMP). Point in the CICS code at which CICS monitoring data is collected. You cannot relocate these system-defined points.

exception class data. CICS monitoring information on exception conditions raised by a transaction, such as queuing for VSAM strings or waiting for temporary storage. This data highlights possible problems in system operations.

exception trace entry. An entry made to the internal trace table and any other active trace destinations when CICS detects an exception condition. It gives information about what was happening at the time the failure occurred and what was being used.

exclusive-key storage. In MVS key-controlled storage protection, storage with storage keys other than open-key.

EXEC. Key word used in CICS command language. All CICS commands begin with the keywords `EXEC CICS`.

Execution Diagnostic Facility (EDF). A facility used for testing application programs interactively online, without making any modifications to the source program or to the program preparation procedure. The facility intercepts execution of the program at various points and displays information about the program at these points. Also displayed are any screens sent by the user program, so that the programmer can converse with the application program during testing just as a user would do on the production system.

extended CICS dynamic storage area (ECDSA). Storage area allocated above the 16MB line for CICS code and control blocks which are eligible to reside above the 16MB line, but are not eligible for the ERDSA (that is, they are not reentrant.) See the *CICS/ESA System Definition Guide* for more information.

extended common system area (ECSA). A major element of MVS/ESA virtual storage above the 16MB line. This area contains pageable system data areas that are addressable by all active virtual storage address spaces. It duplicates the **common system area (CSA)** which exists below the 16MB line.

extended error queue element (EEQE). Contains data describing I/O errors on a database recorded by CICS.

extended link pack area (ELPA). A major element of MVS/ESA virtual storage above the 16MB line. It duplicates the **link pack area (LPA)**. See also **extended addressing**.

extended private area. An element of MVS/ESA virtual storage above the 16MB line. This area duplicates the **private area** except for the 16KB system region area.

extended read-only dynamic storage area (ERDSA). An area of storage allocated above the 16MB line and used for eligible, reentrant CICS and user application programs, which must be link-edited with the RENT and AMODE(31) attributes. The storage is obtained in key 0, non-fetch-protected storage.

Extended Recovery Facility (XRF). A facility that increases the availability of CICS transaction processing, as seen by the end users. Availability is improved by having a second CICS system (the **alternate system**) ready to continue processing the workload, if and when particular failures that disrupt user services occur on the first system (the **active system**).

Extended Recovery Facility (XRF) complex. All the required hardware and software components (MVS images and licensed programs) that provide the XRF function for a CICS system.

extended system queue area (ESQA). A major element of MVS/ESA virtual storage above the 16MB line. This storage area contains tables and queues relating to the entire system. It duplicates above the 16MB line the **system queue area (SQA)**.

extended user dynamic storage area (EUDSA). Storage area allocated above the 16MB line and reserved exclusively for those user application programs that execute in user-key, that are eligible to reside

above the 16MB line, but are not eligible for the ERDSA (that is, they are not reentrant.)

external response time. Elapsed time from pressing the ENTER key or another AID key until the action requested by the terminal user is completed, and the next entry can be started.

external security manager (ESM). A program, such as RACF, that performs security checking for CICS users and resources.

external throughput rate (ETR). The amount of useful work completed in a unit of time (for example, the number of transactions completed per elapsed second).

extrapartition transient data. A CICS facility for temporarily saving data in the form of queues, called destinations. Each extrapartition TD destination requires a resource definition that links it to a QSAM data set outside the CICS region. Each extrapartition TD destination uses a different QSAM data set. Extrapartition destinations are used for data that is either coming from a source outside the region, or being directed from a source within the region to a destination outside the region. Extrapartition data written by CICS is usually intended for subsequent input to non-CICS batch programs. Examples of data that might be written to extrapartition destinations include logging records, statistics, and transaction error messages. Contrast with **intrapartition transient data**.

F

FCT. System initialization parameter used to specify the suffix of the file control table to be used. This parameter is effective only on a CICS cold start. See the *CICS/ESA System Definition Guide* for more information.

file control table (FCT). Table containing the characteristics of the files accessed by file control.

file request thread element (FRTE). An element used by CICS file control to link related requests together as a file thread; to record the existence of READ SET storage to be released at syncpoint and the existence of any other outstanding work that must be completed at syncpoint; to register a task as a user of a file to prevent it being closed while still in use.

file-owning region (FOR). A CICS address space whose primary purpose is to manage files and databases. Deprecated term for **data-owning region (DOR)**. See also **application-owning region (AOR)**, and **terminal-owning region (TOR)**.

first failure data capture (FFDC). Data relevant to a CICS exception condition that is recorded as soon as possible after the condition has been detected.

fixed-block-architecture (FBA) device. A disk storage device that stores data in blocks of fixed size. These blocks are addressed by block number relative to the beginning of the particular file.

format. The arrangement or layout of data on a data medium, usually a display screen with CICS.

format independence. The ability to send data to a device without having to be concerned with the format in which the data is displayed. The same data may appear in different formats on different devices.

fragmentation. The breaking up of free storage into small areas (by intervening used storage areas). This leads to the effective storage available for use being reduced.

FREEMAIN. EXEC CICS command used to release main storage. For programming information, see the *CICS/ESA Application Programming Reference*.

function shipping. The process, transparent to the application program, by which CICS accesses resources when those resources are actually held on another CICS system.

G

GTF. Generalized trace facility – a data-collection routine in MVS. GTF traces the following system events: seek addresses on start I/O records, SRM activity, page faults, I/O activity, and supervisor services. Execution options specify the system events to be traced.

H

high performance option (HPO). A option provided with MVS to improve performance by reducing the transaction pathlength; that is, the number of instructions needed to service each request.

high private area. Part of the CICS address space, consisting of the local system queue area (LSQA), the scheduler work area (SWA), and subpools 229 and 230. The area at the high end of the CICS address space is not specifically used by CICS, but contains information and control blocks that are needed by the operating system to support the region and its requirements.

Hiperspace. A high-performance storage area in the processor or multiprocessor.

host computer. The primary or controlling computer in a data communication system.

host processor. The primary or controlling computer in a multiple computer installation.

HPO. System initialization parameter used to indicate whether you want to use the VTAM authorized path feature of the high performance option. The default is NO. You can code this parameter only in the system initialization table. See the *CICS/ESA System Definition Guide* for more information.

I

I/O. Input/output (primarily from and to terminals)

ICP. System initialization parameter used to specify that you want to cold start the CICS interval control program. See the *CICS/ESA System Definition Guide* for more information.

ICV. System initialization parameter used to specify the region exit interval time in milliseconds. The region exit interval is the maximum interval of time for which CICS releases control to the operating system in the event that there are no transactions ready to resume
processing. The keyword for ICV is time. The default interval is 1000 milliseconds. See the *CICS/ESA System Definition Guide* for more information.

ICVR. System initialization parameter used to specify the runaway task time interval in milliseconds as a decimal number. CICS purges a task if it has not given up control after this length of time (that is, if the task appears to be looping). The default value is 5000 milliseconds. See the *CICS/ESA System Definition Guide* for more information.

ICVTSD. System initialization parameter used to specify the terminal scan delay. The terminal scan delay facility determines how quickly CICS deals with some terminal I/O requests made by applications. The default value is 500 milliseconds. See the *CICS/ESA System Definition Guide* for more information.

in-flight task. (1) A task that is in progress when a CICS system failure or immediate shutdown occurs. (2) During emergency restart, a task that caused records to be written to the system log, but for which no syncpoint record has been found for the current LUW. This task was interrupted before the LUW completed.

in-flight transaction. Any transaction that was still in process when system termination occurred.

Information Management System (IMS). A database manager used by CICS to allow access to data in DL/I databases. IMS provides for the arrangement of data in an hierarchical structure and a common access approach in application programs that manipulate IMS databases.

initialization. (1) Actions performed by the CICS system to construct the environment in the CICS region to enable CICS applications to be run. (2) The stage of the XRF process when the active or the alternate CICS system is started, signs on to the control data set, and begins to issue its surveillance signal.

initialization phase. The process of bringing up the active CICS system and the alternate CICS system in an XRF complex. The two actions are performed independently.

installation. (1) A particular computing system, in terms of the work it does and the people who manage it, operate it, apply it to problems, service it, and use the work it produces. (2) The task of making a program ready to do useful work. This task includes generating a program, initializing it, and applying any changes to it.

interactive. Pertaining to an application in which each entry entails a response from a system or program, as in an inquiry system or an airline reservation system. An interactive system may also be conversational, implying a continuous dialogue between the user and the system.

intercommunication facilities. A term covering intersystem communication (ISC) and multiregion operation (MRO).

intermediate routing node (IRN). A subarea node, which may receive and route sessions that neither originate in nor are destined for network addressable units in that subarea node. Terminals attached to an IRN cannot have XRF backup sessions.

internal response time. Elapsed time from the message to start a transaction being received by CICS until the time that the transaction ends.

internal throughput rate (ITR). The number of completed transactions per processor-busy second. (Processor busy seconds can be calculated by multiplying elapsed seconds by the processor utilization percentage).

internal trace. CICS trace facility that is always present in virtual storage. When CICS detects an exception condition, an entry always goes to the internal trace table, even if you have turned tracing off. The internal trace table is a wraparound table whose size can be set by the TRTABSZ system initialization parameter and can be changed by the CICS SET TRACE DEST command. See the *CICS/ESA Problem Determination Guide* for more information.

interregion communication (IRC). The method by which CICS provides communication between a CICS region and another region in the same processor. Used for **multiregion operation (MRO)** . Compare with **intersystem communication**.

intersystem communication (ISC). Communication between separate systems by means of SNA networking facilities or by means of the application-to-application facilities of VTAM. ISC links CICS systems and other systems, and may be used for user application to user application communication, or for transparently executing CICS functions on a remote CICS system. Compare with **multiregion operation** and **interregion communication**.

interval control element (ICE). An element created for each time-dependent request received by the interval control program. These ICEs are logically chained to the CSA in expiration time-of-day sequence.

Expiration of a time-ordered request is detected by the expired request logic of the interval control program running as a CICS system task whenever the task dispatcher gains control. The type of service represented by the expired ICE is initiated, providing all resources required for the service are available, and the ICE is removed from the chain. If the resources are not available, the ICE remains on the chain and another attempt to initiate the requested service is made the next time the task dispatcher gains control.

interval control program (ICP). The CICS program that provides time-dependent facilities. Together with task control, interval control (sometimes called time management) provides various optional task functions (system stall detection, runaway task control, task synchronization, etc.) based on specified intervals of time, or the time of day.

interval statistics. CICS statistics gathered during a specified interval. See also **end-of-day statistics**, **requested statistics**, **requested reset statistics**, and **unsolicited statistics**.

intrapartition transient data (TD). A CICS facility for temporarily saving data in the form of queues, called destinations. All intrapartition TD destinations are held as queues in the same VSAM data set, which is managed by CICS. Data is written to the queue by a user task. The queue can be used subsequently as input data by other tasks within the CICS region. All access is sequential, governed by read and write pointers. Once a record has been read it cannot be read subsequently by another task. An intrapartition destination requires a resource definition containing information that locates the queue in the intrapartition data set. Applications that might use intrapartition queues include message switching, data collection, and queuing of orders.

ISC. System initialization parameter used to include the CICS programs required for interregion or intersystem communication. See the *CICS/ESA System Definition Guide* for more information.

J

JCT. System initialization parameter used to specify the suffix of the journal control table to be used. This indicates whether journaling and volume control are to be used. See the *CICS/ESA System Definition Guide* for more information.

job control language (JCL). Control language used to describe a job and its requirements to an operating system.

journal. A set of one or more data sets to which records are written during a CICS run:

1. By CICS to implement user-defined resource protection (logging to the system log)
2. By CICS to implement user-defined automatic journaling (to any journal, including the system log)
3. Explicitly by the WRITE JOURNALNUM command from an application program (user journaling to any journal, including the system log).

journal archive control data set (JACD). CICS system data set for use by the CICS automatic journal archive facility to store information about the journal data sets.

journal control. The CICS module that processes logging or journaling requests by the system or the user, and writes this information to the system log or to a journal.

journal control table (JCT). A table in which the system log and user journals and their characteristics are described to CICS for access through journal control. The JCT contains control information and operating system control blocks describing each journal.

K

key 0. Storage used by CICS for the extended read-only dynamic storage area (ERDSA). The allocation of key 0 storage for the ERDSA is optional.

key-controlled storage protection. An MVS facility for protecting access to storage. Access to key-controlled storage is permitted only when the **storage key** matches the **access key** associated with the request.

keypoint. The periodic recording of system information and control blocks on the system log–also the data so

recorded. See also **activity keypoint**, and **warm keypoint**.

L

Last-in-first-out (LIFO). A queuing technique in which the next item to be retrieved is the last item placed in the queue.

LIFO storage. Storage used by reentrant CICS management modules to save registers.

link pack area (LPA). A major element of MVS/ESA virtual storage below the 16MB line. The storage areas that make up the LPA contain all the common reentrant modules shared by the system, and exists to provide economy of real storage by sharing one copy of the modules, protection because LPA code cannot be overwritten even by key 0 programs, and reduced pathlength because the modules can be branched to. The LPA is duplicated above the 16MB line as the **extended link pack area (ELPA)**.

LISTCAT. A VSAM tool that provides information that interprets the actual situation of VSAM data sets.

local. (1) In data communication, pertaining to devices that are accessed directly without use of a telecommunication line. Contrast with **remote**. (2) Synonym for channel-attached.

local DL/I. DL/I residing in the CICS address space.

local resource. In CICS intercommunication, a resource that is owned by the local system.

local shared resources (LSR). Files that share a common pool of buffers and a common pool of strings; that is, control blocks supporting the I/O operations. Contrast with **nonshared resources**.

local system. The system in a multisystem environment on which the application program is executing. The local application may process data from databases located on both the same (local) system and another (remote) system. Contrast with **remote system**.

local system queue area (LSQA). An element of the CICS address space. It generally contains the control blocks for storage and contents supervision. See also **high private area**.

local work area. Area provided for the use of a single task-related user exit program. It is associated with a single task and lasts for the duration of the task only.

log. A recording of changes made to a file. This recording can be used for subsequent recovery of the file. See also **dynamic log**, **journal**, and **system log**.

logging. The recording (by CICS) of recovery information onto the system log, for use during emergency restart. A specific journaling function that records changes made to the system activity environment and database environment. These records are required for recovery and backout support by CICS (and the user) following an abnormal termination.

logical partition (LP). A partition, in a CEC, capable of running its own MVS image. It comprises a set of hardware resources (processors, storage, channels, and so on), sufficient to allow a system control program such as MVS to be executed.

logical unit (LU). In SNA, a port through which a user gains access to the services of a network.

logical unit of work (LUW). A sequence of processing actions (database changes, for example) that must be completed before any of the individual actions can be regarded as committed. When changes are committed (by successful completion of the LUW and recording of the syncpoint on the system log), they do not need to be backed out after a subsequent failure of the task or system. The end of an LUW is marked in a transaction by a syncpoint, issued either by the user program or by CICS at the end of task. In the absence of user syncpoints, the entire task is an LUW.

look-aside query. Query performed in one partition by an operator working in another partition. Using partitions, a partially completed operation need not be transmitted to the host processor before releasing the screen for an inquiry.

LPA. System initialization parameter used to indicate whether any CICS management modules can be used from the link pack area. The default is NO. See the *CICS/ESA System Definition Guide* for more information.

LUTYPE6.1 (LU6.1). Type of logical unit used for processor-to-processor sessions. LUTYPE6.1 is a development of LUTYPE6. CICS–IMS intercommunication uses LUTYPE6.1 sessions.

LUTYPE6.2 (LU6.2). Type of logical unit used for CICS intersystem (ISC) sessions. LUTYPE6.2 is a development of LUTYPE6.1. The LUTYPE6.2 architecture supports both CICS host to system-level products and CICS host to device-level products. CICS ISC uses LUTYPE6.2 sessions. APPC is the the protocol boundary of the LU6.2 architecture.

M

main storage. (ISO) Program-addressable storage from which instructions and data can be loaded directly into registers for subsequent execution or processing. See also **real storage, storage, virtual storage**.

map. A format established for a page or a portion of a page, or a set of screen format descriptions. A concept of CICS BMS that maps the relationship between the names of program variables and the position in which their values will appear on a display device. The map also contains other formatting information such as field attributes. A map describes constant fields, and their position on the display; the format of input and output fields; the attributes of constant and variable fields and the symbolic names of variable fields.

message control program (MCP). In ACF/TCAM, a specific implementation of an access method, including I/O routines, buffering routines, activation and deactivation routines, service facilities, and SNA support.

message data set. In XRF, a data set used by the active CICS system to transmit messages to the alternate CICS system.

message performance option. The improvement of ISC performance by eliminating syncpoint coordination between the connected systems.

message switching. In a data network, the process of routing messages by receiving, storing, and forwarding complete messages.

mirror task. A task required to service any incoming request that specifies a CICS mirror transaction (CSMI, CSM1, CSM2, CSM3, CSM5, CPMI, CVMI).

mirror transaction. Recreates the request that is function shipped from one system to another, issues the request on the second system, and passes the acquired data back to the first system.

MN. System initialization parameter used to indicate whether monitoring is to be switched on or off at initialization. The default is OFF. See the *CICS/ESA System Definition Guide* for more information.

MNEVE. System initialization parameter used to indicate whether SYSEVENT monitoring is to be made active during CICS initialization. The default is OFF. See the *CICS/ESA System Definition Guide* for more information.

MNEXC. System initialization parameter used to indicate whether the monitoring exception class is to be made active during CICS initialization. The default is OFF. See the *CICS/ESA System Definition Guide* for more information.

MNPER. System initialization parameter used to indicate whether the monitoring performance class is to be made active during CICS initialization. The default is OFF. See the *CICS/ESA System Definition Guide* for more information.

modegroup. A VTAM LOGMODE entry, which can specify (among other things) the class of service required for a group of APPC sessions.

modename. The name of a modeset and of the corresponding modegroup.

modeset. In CICS, a group of APPC sessions. A modeset is linked by its modename to a modegroup (VTAM LOGMODE entry) that defines the class of service for the modeset.

modified link pack area (MLPA). An element of MVS/ESA virtual storage. This area provides a temporary extension to the PLPA existing only for the life of the current IPL. You can use this area to add or replace altered LPA-eligible modules without having to recreate the LPA. See also **link pack area (LPA)** and **pageable link pack area (PLPA)**.

monitoring. (1) The regular assessment of an ongoing production system against defined thresholds to check that the system is operating correctly. (2) Running a hardware or software tool to measure the performance characteristics of a system. Note that CICS distinguishes between monitoring and statistics, but IMS does not. See also **statistics**.

monitoring control table (MCT). A table describing the way the user data fields in the accounting and performance class monitoring records are to be manipulated at each user event monitoring point (EMP). It also identifies the CICS user journals in which the data for each monitoring class is to be recorded. The MCT contains the definition of user event monitor points (EMPs), and specifies the journal data sets used to record the data.

monitoring record. Any of three types of task-related activity record (performance, event, and exception) built by the CICS monitoring domain. Monitoring records are available to the user for accounting, tuning, and capacity planning purposes.

MROBTCH. System initialization parameter used to specify the number of events that must occur before CICS is posted for dispatch due to the batching mechanism. The default is one. See the *CICS/ESA System Definition Guide* for more information.

MROLRM. System initialization parameter used to specify whether you want to establish an MRO

long-running mirror task. The default is NO. See the *CICS/ESA System Definition Guide* for more information.

multiregion operation (MRO). Communication between CICS systems in the same processor without the use of SNA network facilities. This allows several CICS systems in different regions to communicate with each other, and to share resources such as files, terminals, temporary storage, and so on. Contrast with **intersystem communication**.

multitasking. Concurrent execution of application programs within a CICS region.

multithreading. Use, by several transactions, of a single copy of an application program.

MVS image. A single copy of the MVS operating system. This can be a physical processing system (such as an IBM 3090) that is partitioned into one or more processors, where each partition is capable of running under the control of a single MVS operating system. Alternatively, if you are running MVS with the processor resource/systems manager (PR/SM), an MVS image can consist of multiple logical partitions, with each logical partition (LP) operating a copy of MVS. Also referred to as a single- or multi-MVS environment, according to the number of MVS systems.

MVS/DFP. MVS/Data Facility Product, a major element of MVS, including data access methods and data administration utilities.

MVS/ESA extended nucleus. A major element of MVS/ESA virtual storage. This area duplicates above the 16MB line the **MVS/ESA nucleus**.

MVS/ESA nucleus. A major element of MVS/ESA virtual storage. This static storage area contains control programs and key control blocks. The area includes the nucleus load module and is of variable size, depending on the installation's configuration. The nucleus is duplicated above the 16MB line as the **MVS/ESA extended nucleus**.

MXT. System initialization parameter used to specify the maximum number of tasks that CICS allows to exist at any time. The default is 32. See the *CICS/ESA System Definition Guide* for more information.

N

NEB. Node error block.

NEP. Node error program.

NETNAME (netname). In CICS/ESA, the name by which a CICS terminal or a CICS system is known to ACF/VTAM.

NetView. A network management product that can provide rapid notification of events and automated operations.

NetView Performance Monitor (NPM). A program product that collects and reports on data in the host and NCP.

network. (1) An interconnected group of nodes. (2) The assembly of equipment through which connections are made between data stations.

network configuration. In SNA, the group of links, nodes, machine features, devices, and programs that make up a data processing system, a network, or a communication system.

network control program (ACF / NCP). A program that controls the operation of a communication controller (3745, 3725, 3720, 3705) in which it resides. NCP builds the backup sessions to the alternate CICS system for XRF-capable terminals. NCP is generated by the user from a library of IBM-supplied modules.

Network Logic Data Manager (NLDM). A program that collects and interprets records of errors detected in a network and suggests possible solutions. NLDM consists of commands and data services processors that comprise the NetView software monitor component.

Network Performance Analysis and Reporting System (NETPARS). A program offering that analyzes network log data from the NetView Performance Monitor (NPM).

Network Problem Determination Application (NPDA). A program that collects and interprets records of errors detected in a network and suggests possible solutions. NPDA consists of commands and data services processors that comprise the NetView hardware monitor component.

NEWSIT. System initialization parameter used to cause CICS to load the specified system initialization table (SIT) and enforce the use of all SIT parameters, modified by any system initialization parameters provided through PARM, SYSIN, or the system console. You can code NEWSIT on PARM, SYSIN, or CONSOLE only. See the *CICS/ESA System Definition Guide* for more information.

node abnormal condition program (NACP). A CICS program used by terminal control to analyze terminal abnormal conditions that are logical unit or node errors detected by VTAM.

node error block (NEB). A set of recording areas of the node error table used to count node errors relating to a single logical unit.

node error program (NEP). A user-replaceable program used to allow user-dependent processing whenever a communication error is reported to CICS

nonconversational. A mode of CICS operation in which resources are allocated, used, and released immediately on completion of the task.

O

operating system. Software that controls the execution of programs; an operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

Operating System/Virtual Storage (OS/VS). A compatible extension of the IBM System/360 Operating System that supports relocation hardware and the extended control facilities of System/360.

OS/VS. Operating System/Virtual Storage.

P

page. (1) In MVS/ESA, a fixed-length block that has a virtual address and that is transferred as a unit between real storage and auxiliary storage. (2) The information displayed at the same time on a display device.

pageable link pack area (PLPA). An element of MVS/ESA virtual storage. This area contains supervisor call routines, access methods, and other read-only system programs along with read-only reenterable user programs selected by an installation to be shared among users of the system. Optional functions or devices selected by an installation during system generation add additional modules to the PLPA. See also **link pack area (LPA)** and **modified link pack area (MLPA)**.

paging. In MVS, the process of transferring pages between real storage and the external page storage known as the page data set.

performance. Together with ease-of-use (a measure of how easy it is to use a data processing system), a major factor on which the total productivity of a system depends. Performance is largely determined by a combination of three other factors: availability, response time, and throughput, and reflects the overall quality of service and operations of a given product or system.

performance analysis. The use of one or more performance tools to investigate the reasons for performance deterioration.

performance class data. Detailed transaction-level monitoring data, such as the processor and elapsed time for a transaction, or the time spent waiting for an I/O.

performance data section. Part of the CICS data section in a CICS monitoring record. It consists of a string of field connectors followed by one or more performance data records.

performance evaluation. The determination of how well a specific system is meeting or may be expected to meet specific processing requirements at specific interfaces. Performance evaluation, by determining such factors as throughput rate, turnaround time, and constrained resources, can provide important inputs and data for the performance improvement process.

performance improvement. The increase of the average throughput rate and operational capability, or the reduction of turnaround time.

persistent verification (PV). The retention of a sign-on to a remote system across multiple conversations until it is no longer needed. The PVDELAY system initialization parameter defines how long entries can remain in the signed-on list in the remote system. See **PVDELAY**.

PISCHD. System initialization parameter used to specify whether program isolation scheduling or segment intent scheduling is to be performed for transactions that access CICS local DL/I databases. See the *CICS/ESA System Definition Guide* for more information.

PL/I. A programming language designed for use in a wide range of commercial and scientific applications.

planned takeover. In XRF, a planned shutdown of the active CICS system, and takeover by the alternate system, for maintenance or operational reasons.

polling. The process whereby stations are invited, one at a time, to transmit. The polling process usually involves the sequential interrogation of several data stations.

post-takeover. The XRF phase, immediately following takeover, when the new active CICS system does not have an alternate system.

pregenerated system. A CICS system distributed in a form that has already undergone the system generation process.

priority. A rank assigned to a task that determines its precedence in receiving system resources.

private area. A major element of MVS/ESA virtual storage below the 16MB line. It contains the local system queue area (LSQA), scheduler work area, subpools 229 and 230, a 16KB system region area, and

a private user region for running programs and storing data. This area is duplicated (except for the 16KB system region area) above the 16MB line as the **extended private area**.

profile. (1) In CICS, a set of options specified in a resource definition that can be invoked by a transaction definition. Profiles control the interactions between the transaction and terminals or logical units. CICS supplies profile definitions suitable for most purposes. If a transaction definition does not specify a profile, a standard profile is used. (2) In RACF, data that describes the significant characteristics of a user, a resource, a group of users, or a group of resources. See **resource profile**, **discrete profile**, **generic profile**, **user profile**, **resource group profile**, **data set profile**.

program check. A condition that occurs when programming errors are detected by a processor during execution.

program communication block (PCB). IMS control block that describes an application program's interface to an IMS database or, additionally, for message processing and batch message processing programs, to the source and destination of messages. See also **program specification block (PSB)**.

program compression. An operation performed by program control to relieve space in the DSA during a short-on-storage condition. The PPT is searched to identify programs that have been dynamically loaded and are currently not in use. If a program is not in use, the space it occupied is reclaimed.

program isolation (PI). An IMS facility that protects all activity of an application program from any other active application program until that application program indicates, by reaching a syncpoint, that the data it has modified is consistent and complete.

PRTYAGE. System initialization parameter used to specify the number of milliseconds to be used in the priority aging algorithm for incrementing the priority of a task. The default is 32 768 milliseconds. See the *CICS/ESA System Definition Guide* for more information.

PRVMOD. System initialization parameter used to specify the names of those modules that are not to be used from the LPA. See the *CICS/ESA System Definition Guide* for more information.

PSB directory (PDIR). A list or directory of program specification blocks (PSBs) that define for DL/I the use of data bases by application programs. It contains one entry for each PSB to be used during CICS execution, and is loaded during initialization. Each entry contains the size of the control block, the status, the storage location (if in storage), and the DASD address of the PSB in the ACBLIB. It is generated using DFHDLPSB macros. Contains entries defining each PSB to be accessed using local DL/I. Also contains entries for remote PSBs, to which requests are function-shipped using remote DL/I.

PSBCHK. System initialization parameter used to request DL/I security checking of a remote terminal initiating a transaction with transaction routing. This parameter is only applicable if the local CICS-DL/I interface is being used. The default is to have the remote link checked but no check made against the remote terminal.

PSBPL. System initialization parameter used to specify the size of the PSB pool in 1024-byte blocks for local CICS-DL/I interface support. This parameter is only applicable if the local CICS-DL/I interface is being used. The default is four blocks.

pseudoconversational. A type of CICS application design that appears to the user as a continuous conversation, but that consists internally of multiple tasks — also called "transaction-oriented programming."

purge. The abending of a task by task control to alleviate a short-on-storage condition.

PUT. Program update tape.

PVDELAY. System initialization parameter used define how long entries can remain in the PV signed-on-from list on the remote system. The default is 30 minutes. See **persistent verification**. See the *CICS/ESA System Definition Guide* for more information.

Q

QSAM. Queued sequential access method.

quasi-reentrant. Applied to a CICS application program that is serially reusable between entry and exit points because it does not modify itself or store data within itself between calls on CICS facilities.

queue. A line or list formed by items in a system waiting for service; for example, tasks to be performed, or messages to be transmitted in a message-switching system. In CICS, the transient data and temporary storage facilities store data in queues. See **temporary storage (TS)**, **transient data (TD)**.

queued sequential access method (QSAM). An extended version of BSAM that incorporates queues of input and output blocks that are awaiting processing and transfer respectively.

R

RACF. The Resource Access Control Facility program product. An external security management facility available under MVS.

RAIA. Receive-any input area.

RAMAX. System initialization parameter used to specify the size in bytes of the I/O area allocated for each RECEIVE ANY issued by CICS. The default is 256 bytes. See the *CICS/ESA System Definition Guide* for more information.

RAPOOL. System initialization parameter used to determine the size of the CICS receive-any pool, which is set aside for VTAM receive-any operations. See the *CICS/ESA System Definition Guide* for more information.

real storage. The main storage in a virtual storage system. Physically, real storage and main storage are identical. Conceptually, however, real storage represents only part of the range of addresses available to the user of a virtual storage system.

receive-any control element (RACE). Type of control field held in the CICS receive-any pool set aside for VTAM receive-any operations. The number of RACEs maintained depends on the RAPOOL and MXT system initialization parameters and on the number of active tasks. See the *CICS/ESA System Definition Guide* for more information.

receive-any input area (RAIA). Type of input area held in the CICS receive-any pool set aside for VTAM receive-any operations. The number of RACEs maintained depends on the RAPOOL and MXT system initialization parameters and on the number of active tasks. See the *CICS/ESA System Definition Guide* for more information.

recoverability. The ability of a system to continue processing without loss of data when an unplanned interruption occurs.

recoverable in-doubt structure (RIS). In DBCTL, an area constructed for each unit of recovery when a failure occurs. Each RIS is written to the IMS log. RIS contents include the recovery token, the changed data records, and the identity of the data block that cannot be accessed because of unresolved in-doubts.

recovery. The process of returning the system to a state from which operation can be resumed. The restoration of resources following an error.

recovery manager. CICS resource recovery mechanism that provides a CICS resource manager, for example file control, with more flexibility than the DWE two-phase commit support for syncpoint and backout processing.

reference set. The amount of real storage required so that minimal (almost zero) virtual paging occurs. It is the total amount of real storage required to process the most frequently used sequence of instructions and data for a given set of transactions performing defined tasks, without causing any virtual storage paging operations.

region. A section of the dynamic area that is allocated to a job step or system task. The term is used to cover partitions and address spaces in addition to regions.

region-remote. A term used in early releases of CICS to refer to a CICS system in another region of the same processor. It can be taken to refer to a system that is accessed through an IRC (MRO) link, rather than through an SNA LU6.1 or LU6.2 link.

reliability. A measurement of the ability of a system to continue processing without failure. Shutting down an on-line system to process batch updates to the database subtracts from its availability to end users, but this has no bearing on reliability of components required to deliver the online service.

remote. In data communication, pertaining to devices that are connected to a data processing system through a data link. Synonym of link-attached. Contrast with **local**.

remote DL/I. A special case of function shipping, in which CICS sends a DL/I request to another CICS system. See also **function shipping**.

remote resource. In CICS intercommunication, a resource that is owned by a remote system. Contrast with **local resource**.

remote system. In CICS intercommunication, a system that the local CICS system accesses via intersystem communication or multiregion operation. Contrast with **local system**.

RENTPGM. System initialization parameter used to specify whether CICS is to allocate the extended read-only dynamic storage area (ERDSA) from read-only, key 0 protected storage (the default), or from CICS-key storage. Specifying CICS-key storage effectively creates a second ERDSA and allows ERDSA-eligible programs that execute in CICS-key to modify storage where required.

request parameter list (RPL). In ACF/VTAM, a control block that contains the parameters necessary for processing a request for data transfer, for connecting or disconnecting a terminal, or for some other operation.

requested reset statistics. CICS statistics that the user has asked for by using the appropriate EXEC

CICS or CEMT commands, which cause the statistics to be written to the SMF data set immediately. Requested reset statistics differ from requested statistics in that the statistics counters are reset, using an EXEC CICS or CEMT command.

requested statistics. CICS statistics that the user has asked for by using the appropriate EXEC CICS or CEMT commands, which cause the statistics to be written to the SMF data set immediately, instead of waiting for the current interval to expire. Contrast with **requested statistics**.

residence mode (RMODE). Attribute of a program indicating where it can reside, that is, either above or below the 16MB line.

resource. Any facility of the computing system or operating system required by a job or task, and including main storage, input/output devices, the processing unit, data sets, and control or processing programs.

Resource Access Control Facility (RACF). An IBM licensed product that provides for access control by identifying and verifying users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, and logging detected accesses to protected resources.

resource control table (RCT). A control table that defines the relationship between CICS transactions and DB2 resources. For details, refer to the *DB2 Version 2 Administration Guide*.

resource measurement facility (RMF). An IBM program that collects system-wide data describing the processor activity (WAIT time), I/O activity (channel and device utilization), main storage activity (demand and swap paging statistics), and system resources manager (SRM) activity (workload). RMF produces two types of report, system-wide reports and address-space reports.

response time. The elapsed time from entry of a last input message segment to the first response segment.

restart. Resumption of operation after recovery. Ability to restart requires knowledge of where to start and ability to start at that point.

restart data set (RDS). A VSAM KSDS used only during emergency restart. The RDS temporarily holds the backout information read from the CICS system log. This allows CICS to be restored to a stable state and to be restarted following an abrupt termination.

RMODE. In MVS, an attribute that specifies the residence mode of the load module produced by the linkage editor. Unless a program is link-edited with RMODE(24), CICS/ESA loads it above the 16MB line if possible.

RMTRAN. System initialization parameter used with XRF to specify the name of the transaction that you want an alternate CICS to initiate when logged-on class 1 terminals are switched following a takeover. This parameter is applicable only on an alternate CICS region.

rollback. A programmed return to a prior checkpoint. In CICS, the cancellation by an application program of the changes it has made to all recoverable resources during the current logical unit of work.

rotational position sensing (RPS). A feature that permits a disk storage device to disconnect from a block multiplexer channel (or its equivalent), allowing the channel to service other devices on the channel during positional delay.

RPL. See **request parameter list**.

RPS. See **rotational position sensing**.

RSD (restart data set). The direct-access data set used to contain the information necessary to restart CICS.

S

SAM. Sequential access method.

sample statistics program (DFH0STAT). IBM-supplied batch program that provides information that is useful in calculating the storage requirements of a CICS/ESA system, for example, the sizes of the dynamic storage areas.

SAS (single address space). Single CICS region; usually used when contrasting with MRO.

scheduler work area (SWA). An element of the CICS address space. The SWA is made up of subpools 236 and 237 which contain information about the job and the step itself. Almost anything that appears in the job stream for the step creates some kind of control block in this area.

SCS. SNA character string.

SCS. System initialization parameter used to specify how much of the dynamic storage area (DSA) you want CICS to regard as the DSA storage cushion. The default is 64KB.

SDB. Structured database.

SDF. Screen Definition Facility. An online application development program product used to define or edit BMS maps interactively.

security. Prevention of access to or use of data or programs without authorization.

segment search argument (SSA). Segment search arguments (SSAs) are used to identify segments of a DL/I database. SSAs may be simple segment names or they may be qualified to include constraints made upon the values of fields within the named segment types. Except for a read-only operation, when it is unnecessary, SSAs used by a CICS application program must be in dynamic storage because of the requirement for the program to be quasi-reenterable.

sequential access method (SAM). An access method for storing and retrieving data blocks in a continuous sequence. Queued sequential access method (QSAM) extends the basic sequential access method (BSAM) by queuing the input and output blocks.

sequential data set. A data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape.

service elements. The discrete hardware and software products that provide a terminal user with processing ability.

Service Level Reporter (SLR) Version 3. A program product that produces reports on CICS performance and service levels. The reports can be used for performance management and many other purposes.

session recovery. The process in which CICS switches active sessions on class 1 terminals to backup sessions or reestablishes service on class 2 terminals.

short-on-storage (SOS). The condition in CICS that occurs when requests for storage from the dynamic storage areas exceed available storage. CICS cannot satisfy these requests, or can satisfy them only by using some of the storage cushions, even when all programs that are eligible for deletion, and are not in use, have been deleted. See also **storage cushion** and **program compression**.

single threading. The execution of a program to completion. Processing of one transaction is completed before another transaction is started. (Compare this with **multithreading**.)

SIP. CICS system initialization program.

SIT. System initialization parameter used to specify the suffix, if any, of the system initialization table (SIT) that you want CICS to load at the start of initialization. If you omit this parameter, CICS loads the pregenerated, default SIT, DFHSIT$$.

SLR. Service Level Reporter.

SMF. System management facilities.

SMF header. Describes the system creating the output.

SMF product section. Component of a CICS monitoring SMF record. It identifies the subsystem to which the monitoring data relates, which, in the case of CICS monitoring and statistics, is the CICS region.

SNA. Systems Network Architecture.

SNT. Signon table.

SOS. Short-on-storage.

SOT. CICS start of task.

SQL/DS. Structured Query Language/Data System. A relational database management facility.

SRB. Service request block (MVS).

SRL. System reference library – IBM-provided manuals that describe a programming or hardware product.

SRM. System resources manager – a component of the MVS control program.

SRT. (1) System recovery table. (2) System initialization parameter used to specify the system recovery table suffix. See the *CICS/ESA System Definition Guide* for more information.

SSA. Segment search argument.

startup. The operation of starting up CICS by the system operator.

startup jobstream. A set of job control statements used to initialize CICS.

statistics. System statistics are accumulated continually by CICS management programs in CICS system tables during the execution of CICS. System statistics can be captured and recorded, either on request or automatically at intervals, by any operator whose security code allows access to such information. In addition, system statistics are recorded on normal termination of the system.

See **automatic statistics**, **unsolicited statistics**, **end-of-day statistics**, **requested statistics**, and **requested reset statistics**.

statistics utility program (DFHSTUP STUP). Provides a summary report facility that can be used to interpret CICS statistics.

STATRCD. System initialization parameter used to set the statistics recording status at CICS initialization. The default is OFF; CICS interval and unsolicited statistics

are not collected. End-of-day statistics are collected at the logical end of day and on shutdown. See the *CICS/ESA System Definition Guide* for more information.

STGPROT. System initialization parameter used to specify that storage protection is required in the CICS region. The default is NO: CICS runs in a single storage key. See the *CICS/ESA System Definition Guide* for more information.

STGRCVY. System initialization parameter used to indicate whether CICS should try to recover from a storage violation. The default is NO. See the *CICS/ESA System Definition Guide* for more information.

storage. A functional unit into which data can be placed and from which it can be retrieved. See **main storage, storage, virtual storage.**

storage accounting area (SAA). A field at the start of a CICS storage area that describes the area and enables CICS to detect some storage violations. Each CICS storage area has either an SAA or a storage check zone.

storage control. The CICS element that gets working storage areas.

storage cushion. A noncontiguous area of storage in the dynamic storage areas reserved for use by CICS when processing a short-on-storage condition.

storage dump. See **transaction dump**.

storage key. An indicator associated with each 4KB block of storage that is available in the CICS region. Access to the storage is controlled by **key-controlled storage protection**.

storage protection. A optional facility in CICS/ESA 3.3 that enables users to protect CICS code and control blocks from being overwritten inadvertently by application programs.

storage protection key. An indicator that appears in the current program status word whenever an associated task has control of the system. This indicator must match the storage keys of all main storage blocks that the task is to use.

storage violation. An error in a storage accounting chain in the dynamic storage area. A storage violation can be detected by the storage manager domain.

storage violation dump. A formatted dump taken as a result of a storage error detected by the storage control program, including a dump of the dynamic storage error.

subpool 229. An element of the CICS address space used primarily for the staging of messages. JES uses this area for messages to be printed on the system log and JCL messages as well as SYSIN/SYSOUT buffers.

subpool 230. An element of the CICS address space used by VTAM for inbound message assembly for segmented messages. Data management keeps data extent blocks (DEBs) here for any opened data set.

SUBTSKS. System initialization parameter used to define the number of task control blocks (TCBs) you want CICS to use for running tasks in concurrent mode. A concurrent mode TCB allows CICS to perform management functions such as system subtasks. The default is none. See the *CICS/ESA System Definition Guide* for more information.

summary report. A statistics report produced by the CICS statistics utility program (STUP). It summarizes the interval, unsolicited, requested reset, and end-of-day statistics on an applid by applid basis.

surveillance. In XRF, a series of processes by which the alternate CICS system monitors the active CICS system for a lapse of activity in order to detect potential failure conditions requiring a takeover. The active and alternate CICS systems use the CAVM surveillance mechanism to monitor each other's well-being.

surveillance signal. In XRF, the signal continuously written to the CAVM data sets by the active and alternate CICS systems to inform the each other of their states.

SVC. Supervisor call.

switch data traffic (SWDT). In an XRF configuration, a VTAM session control request sent to the NCP that initiates the switch of LU sessions from backup XRF session status to active XRF session status. The former XRF session, if still 'active', is terminated with an UNBIND. The switch request is issued to VTAM from the application program (alternate CICS system). VTAM passes the request to the boundary network node, where the sessions are actually switched by NCP.

switched connection. A connection that is established by dialing.

synchronization. The stage of the XRF process when the active and the alternate are both initialized, are aware of each other's presence, and the alternate is ready to begin tracking.

synchronization level (sync level). The level of synchronization (0, 1, or 2) established for an APPC session between intercommunicating CICS transactions. Level 0 gives no synchronization support, level 2 allows the exchange of private synchronization requests, and

level 2 gives full CICS synchronization support with backout of all updates to recoverable resources if failure occurs.

synchronization phase. The XRF phase, immediately after initialization, when the alternate system builds the CICS control blocks to mirror those in the active system.

syncpoint. A logical point in execution of an application program where the changes made to the databases by the program are consistent and complete and can be committed to the database. The output, which has been held up to that point, is sent to its destination(s), the input is removed from the message queues, and the database updates are made available to other applications. When a program terminates abnormally, CICS recovery and restart facilities do not backout updates ***prior*** to the last completed syncpoint.

A syncpoint is created by any of the following:

- A DL/I CHECKPOINT command or CHKP call
- A DL/I TERMINATE command or TERM call
- An EXEC CICS SYNCPOINT request
- An end of task or an end of program.

See also **logical unit of work (LUW)**.

SYSEVENT data. A class of monitoring data which provides a special kind of transaction timing information.

SYSGEN. System generation.

system. In CICS, an assembly of hardware and software capable of providing the facilities of CICS for a particular installation.

system activity keypoint. A keypoint written to the system log automatically while CICS is running normally. (See also **activity keypoint**.)

system dump (SDUMP). In CICS, an MVS SDUMP, which may be formatted with a CICS-supplied IPCS exit to show all control blocks and storage areas in the CICS region.

system dump code. A code that identifies a user-defined entry in the system dump table. Each entry defines system actions and dump attributes to be associated with a code. Codes can be up to 8 characters in length.

A code consisting of the last six characters of a CICS message number describes actions to be taken and the dump to be produced when that message is issued. For example, a dump code table entry for ZZxxxx describes the system dump to be produced with message DFHZZxxxx, and overrides the documented system action for DFHZZxxxx.

Every system dump code can be invoked by EXEC CICS PERFORM DUMP SYSTEM commands and the corresponding CEMT transactions. For the definition of system dump code entries, see **system dump table**.

system generation (SYSGEN). In CICS, the process of creating a particular system tailored to the requirements of a data processing installation.

system initialization table (SIT DFHSIT). A CICS table that contains information to initialize and control system functions, module suffixes for selection of user-specified versions of CICS modules and tables, and information used to control the initialization process. You can generate several SITs, using the resource definition macro DFHSIT, and then use the SIT system initialization parameter to select the one that best meets your current requirements at initialization time.

system log. The (only) journal (identification='01') that is used by CICS to log changes made to resources for the purpose of backout on emergency restart.

system management facility (SMF). MVS management program. CICS stores monitoring and statistical data on SMF data sets.

system program. A program providing services in general support of the running of a system.

system queue area (SQA). A major element of MVS/ESA virtual storage below the 16MB line. This storage area contains tables and queues relating to the entire system. Its contents are highly dependent on the configuration and job requirements at installation. The equivalent area above the 16MB line is the **extended system queue area (ESQA)**.

system recovery table (SRT). A table listing the ABEND or abnormal condition codes that CICS will intercept.

system support program. A program product that defines and generates an NCP and provides it with utility programs.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through and controlling the configuration and operation of networks.

T

takeover. In XRF, the shift of the workload from the active to the alternate CICS system, and the switching of resources needed for this to happen.

takeover phase. In XRF, the replacement of the failing active CICS system by the alternate CICS system as the session partner of the CICS users.

takeover time. In XRF, the elapsed time between the occurrence of a failure, the completion of switching all terminals to the alternate CICS system, and the running of the first user transaction.

task. (1) A unit of work for the processor; therefore the basic multiprogramming unit under the control program. (2) Under CICS, the execution of a transaction for a particular user. Contrast with **transaction**.

task control. The CICS element that controls all CICS tasks.

task control block (TCB). An MVS control block. A TCB is created for each MVS task. Several TCBs are created for CICS management programs. All CICS application programs and non-reentrant CICS code run under a single quasi-reentrant TCB.

task switching. Overlapping of I/O operations and processing between several tasks.

TCA. Task control area.

TCAM. Telecommunications access method.

TCP. System initialization parameter used to include the pregenerated non-VTAM terminal control program, DFHTCP.

TCT. System initialization parameter used to indicate which terminal control table, if any, is to be loaded.

TCTTE. Terminal control table terminal entry.

TCTUA. Option of the ADDRESS command, used also to pass information between application programs, but only if the same terminal is associated with the application programs involved (which can be in different tasks). The pointer reference is set to the address of the TCTUA. If a TCTUA does not exist, the pointer reference is set to X'FF000000'. The data area contains the address of the TCTUA of the principal facility, not that for any alternate facility that may have been allocated.

TCTUAKEY. System initialization parameter used to specify the storage key for the TCTUAs if CICS is operating with storage protection. The default is user-key: a user program executing in any key can modify the TCTUA. See the *CICS/ESA System Definition Guide* for more information.

TCTUALOC. System initialization parameter used to include where terminal user areas (TCTUAs) are to be stored. The default is below the 16MB line. See the *CICS/ESA System Definition Guide* for more information.

TD. (1) CICS transient data. (2) System initialization parameter used to specify the number of VSAM buffers and strings to be used for intrapartition transient data. See the *CICS/ESA System Definition Guide* for more information.

Telecommunications Access Method (TCAM). An access method used to transfer data between main storage and remote or local storage.

Teleprocessing Network Simulator (TPNS). A program used to test new functions before they encounter production volumes.

temporary storage (TS). A CICS facility for temporarily saving data in the form of sequential queues. A TS queue is held in main storage or on a VSAM data set on DASD. All queues not in main storage are in a single VSAM data set. A task can create a TS queue with a name selected by the task. The queue exists until deleted by a task (usually, but not necessarily, the task that created it). Compare **transient data**. Possible uses of temporary storage include storage of screen images for terminal paging and storage of incomplete data for suspended tasks. In general, TS queues do not require resource definition, but see **temporary storage table (TST)**.

temporary storage group identification (TSGID). A control block containing entries addressing each element of a temporary storage queue. Each temporary storage queue has at least one TSGID. Extra TSGID entries are allocated as required. See the *CICS/ESA System Definition Guide* for more information.

terminal. (1) In CICS, a device, often equipped with a keyboard and some kind of display, capable of sending and receiving information over a communication channel. (2) A point in a system or communication network at which data can either enter or leave.

terminal control. The CICS modules that control all CICS terminal activity.

terminal control program (TCP). The program that controls all CICS terminal activity.

terminal control table (TCT). CICS control table retained to define non-VTAM terminal networks.

terminal input/output area (TIOA). Area that is set up by storage control and chained to the terminal control table terminal entry (TCTTE) as needed for terminal input/output operations.

terminal list table (TLT). CICS control table that allows terminal, or operator identifications, or both, to be grouped logically. See **supervisory terminal functions**.

terminal paging. A set of commands for retrieving pages of an oversize output message in any order.

terminal-owning region (TOR). A CICS region which owns most or all of the terminals defined locally. See also **application-owning region (AOR)**, **data-owning region (DOR)**.

termination phase. The XRF phase in which the XRF complex returns to two separate and independent environments and all XRF activity in the alternate system stops.

thread. In CICS, a link between a CICS application and DBCTL. To DBCTL, a thread represents the CICS transaction that has issued a DL/I request. The system initialization parameter DLTHRED specifies the number of threads provided through the CICS local DL/I interface.

threading. The process whereby various transactions undergo concurrent execution.

throughput. The total data processing work successfully completed during an evaluation period.

throughput rate. The data processing work successfully completed per unit of time.

TIOA. Terminal input/output area. TIOAs are acquired and chained to the TCTTE as needed for terminal input/output operations. The field TCTTESC addresses the first terminal-class storage area obtained for a task (the beginning of the chain) and the field TCTTEDA gives the address of the active TIOA. CICS terminal control passes data received from a terminal to the CICS application program in the TIOA, and writes data from the TIOA to the terminal.

TLT. Terminal list table.

trace. Facility for recording CICS activity. There are three destinations for trace entries: internal trace, auxiliary trace, and generalized trace facility (GTF).

trace control. The CICS element that provides a trace facility.

tracking. In XRF, the process by which the alternate CICS system mirrors the starting and stopping of terminal sessions in the active CICS system so that it is prepared to take over the active system should the need arise.

transaction backout. The cancelation, as a result of a transaction failure, of all updates performed by a task.

transaction dump. A dump of the control blocks and storage areas associated with a particular task.

transaction identification code. Synonym for transaction identifier. For example, a group of up to four characters entered by an operator when selecting a transaction.

transaction identifier. A name of up to four characters that is specified when the transaction is defined to CICS and and that is used to invoke the transaction. For example, to select a transaction, a terminal operator enters the transaction identifier.

transaction rate. The number of units of processing successfully completed per unit of time.

transaction restart. The restart of a task after a transaction backout.

transient data (TD). A CICS facility for temporarily saving data in the form of queues, called destinations. A TD destination is held as either as a queue in a VSAM data set managed by CICS (intrapartition TD) or as a QSAM data set outside the CICS region. See **intrapartition transient data** and **extrapartition transient data**. Contrast with **temporary storage**.

TRTABSZ. System initialization parameter used to specify the size, in kilobytes, of the internal trace table. The default (and minimum) size is 16KB.

TSGID. Temporary storage group identification control block. Each temporary storage queue has at least one TSGID which contains entries addressing each element of the queue. Extra TSGID entries are allocated as required.

TSMGSET. System initialization parameter used to specify the number of entries for which dynamic storage is allocated for storing pointers to records put to a temporary storage message set. The default is 4. See the *CICS/ESA System Definition Guide* for more information.

TST. System initialization parameter used to specify the suffix of the temporary storage table. See the *CICS/ESA System Definition Guide* for more information.

TSUT. Temporary storage unit table. The TSUT contains an entry for each temporary storage identifier. Each entry addresses either the temporary storage record in main or in auxiliary storage, or, in the case of a temporary storage queue, the TSGID.

tuning. The process of adjusting system control variables to make the system divide its resources most efficiently for the workload.

turnaround time. The elapsed time between entry of the first character of the first input into the input interface and the passage of the last character of the last output through the output interface. The total time

consumed from the start to the completion of a specific unit of work measured at specific interfaces. When multiple inputs or multiple outputs are parts of one unit of work, intermediate turnaround time specifications may be needed.

TWA. (1) Transaction work area. (2) Option of the ADDRESS command, used also to pass information between application programs, but only if they are in the same task. The pointer reference is set to the address of the TWA. If a TWA does not exist, the pointer reference is set to X'FF000000'.

TWX. Teletypewriter exchange terminal.

U

UDSASZE. System initialization parameter used to specify the amount of storage to be allocated by CICS for the user dynamic storage area (UDSA) below the 16MB line. The default is 3MB. See the *CICS/ESA System Definition Guide* for more information.

unsolicited statistics. CICS statistics automatically gathered by CICS for dynamically allocated and deallocated resources. See also **interval statistics**, **end-of-day statistics**, **requested statistics**, and **requested reset statistics**.

unwanted takeover. In XRF, a takeover initiated by the alternate CICS system when there was not an actual failure on the active CICS system. This might be due to an unusual system condition which, although not a true failure, slowed down the active system's participation in the surveillance process to the point where the alternate system believed that a failure on the active system had occurred.

update. To modify a file or data set with current information.

USCS. System initialization parameter used to specify how much of the user dynamic storage area (UDSA) is to be used for the storage cushion. See the *CICS/ESA System Definition Guide* for more information.

user activity keypoint. A keypoint written to the system log by a user transaction. See also **activity keypoint**.

user dynamic storage area (UDSA). A storage area in CICS/ESA 3.3 allocated below the 16MB line and reserved exclusively for those user application programs that execute in user-key and that reside below the 16MB line.

user exit. A point in an IBM-supplied program at which a user exit routine may be given control. For programming information, see the *CICS/ESA Customization Guide*.

user-key. Storage obtained by CICS in MVS open-key storage. It is for user application programs and their associated data areas. It can be accessed and modified by user applications and by CICS. See **CICS-key**, **storage protection**.

V

validity of reference. Direct reference to the required pages, without intermediate storage references that retrieve unwanted data.

virtual machine (VM). A functional simulation of a computer and its associated devices. Contrast with **real machine**.

virtual storage (VS). (ISO) The notional storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computing system and by the amount of auxiliary storage available and not by the number of main storage locations.

Virtual Storage Access Method (VSAM). An access method for direct or sequential processing of fixed- and variable-length records on direct access devices.

virtual storage constraint relief (VSCR). The movement of areas of code or control blocks to storage above the 16MB line, or the reduction of code or control blocks below the 16MB line. These actions increase the storage available for user programs and data that use 24-bit addressing.

virtual storage paging. A technique used by CICS in a virtual storage environment. The key objective of programming in this environment is the reduction of page faults. A page fault occurs when a program refers to instructions or data that do not reside in real storage, in which case, the page in virtual storage that contains the referenced instructions or data must be paged into real storage. The more paging required, the lower the overall system performance.

VLF. The Virtual Lookaside Facility in MVS/ESA. Manages the data space associated with library lookaside (LLA).

VS. See **virtual storage**.

VSAM shared resources. Buffers and strings shared by several VSAM data files. This is defined to CICS in the file control table.

VSAM work area (VSWA). An area that is acquired dynamically by the file control program when accessing a VSAM data set.

VSCR. Virtual storage constraint relief.

VTAM. (1) Virtual telecommunications access method. (2) System initialization parameter used to include the VTAM access method. The default is YES. See the *CICS/ESA System Definition Guide* for more information.

VTAM Performance Analysis and Reporting System (VTAMPARS). A program offering that provides information on network traffic through the VTAM component of the network.

W

warm keypoint. A keypoint written to the restart data set during controlled shutdown (after all system activity has ceased). During a subsequent warm restart, information in the warm keypoint is used to reestablish system tables to the status they had at controlled shutdown. See also **keypoint**.

warm start. (1) Initialization of a CICS system using selected system status information obtained during the previous termination. (2) Automatic restart after a normal (controlled) shutdown.

working set. (1) The set of a user's pages that must be active in order to avoid excessive paging. (2) The amount of real storage required in order to avoid excessive paging.

X

XLT. (1) Transaction list table. CICS control table containing a list of logically-related transaction identifications. (2) System initialization parameter used to specify the suffix for the transaction list table. See the *CICS/ESA System Definition Guide* for more information.

XMOLS. CICS monitor listing program, DFH$MOLS.

XRF (Extended Recovery Facility). Extended Recovery Facility (XRF).

XRF complex. The MVS images and licensed programs that provide an XRF service.

Index

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Sending your comments to IBM

CICS/ESA

Performance Guide

SC33-1183-02

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book. Please limit your comments to the information in this book 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.

You can send your comments to IBM in any of the following ways:

- By mail, use the Readers' Comment Form
- By fax:
  - From outside the U.K., after your international access code use 44 962 870229 (after 16 April 1995, use 44 1962 870229)
  - From within the U.K., use 0962 870229 (after 16 April 1995, use 01962 870229)
- Electronically, use the appropriate network ID:
  - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
  - IBMLink: WINVMJ(IDRCF)
  - Internet: idrcf@winvmj.vnet.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The page number or topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Readers' Comments

CICS/ESA

Performance Guide

SC33-1183-02

Use this form to tell us what you think about this manual. If you have found errors in it, or if you want to express your opinion about it (such as organization, subject matter, appearance) or make suggestions for improvement, this is the form to use.

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. This form is provided for comments about the information in this manual and the way it is presented.

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.

Be sure to print your name and address below if you would like a reply.

Name	Address
Company or Organization	
Telephone	Email

CICS/ESA Performance Guide
SC33-1183-02

You can send your comments POST FREE on this form from any one of these countries:

Australia	Finland	Iceland	Netherlands	Singapore	United States
Belgium	France	Israel	New Zealand	Spain	of America
Bermuda	Germany	Italy	Norway	Sweden	
Cyprus	Greece	Luxembourg	Portugal	Switzerland	
Denmark	Hong Kong	Monaco	Republic of Ireland	United Arab Emirates	

If your country is not listed here, your local IBM representative will be pleased to forward your comments to us. Or you can pay the postage and send the form direct to IBM (this includes mailing in the U.K.).

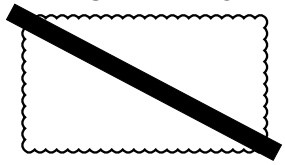
1 Cut along this line

2 Fold along this line

By air mail
Par avion

IBRS/CCRI NUMBER: PHQ - D/1348/SO

NE PAS AFFRANCHIR



NO STAMP REQUIRED

REPONSE PAYEE
GRANDE-BRETAGNE

IBM United Kingdom Laboratories Limited
Information Development Department (MP 095)
Hursley Park
WINCHESTER, Hants
SO21 2ZZ United Kingdom

3 Fold along this line

From: Name _____
Company or Organization _____
Address _____
_____
EMAIL _____
Telephone _____

1 Cut along this line

4 Fasten here with adhesive tape